VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Softwarový systém pro automatizaci a integraci testovacích strojů

Software System for Tester Machine Automation and Integration

2017 Bc. Pavel Šamánek

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání diplomové práce

Student: **Bc. Pavel Šamánek**

Studijní program: N2647 Informační a komunikační technologie

Studijní obor: 2612T025 Informatika a výpočetní technika

Téma: Softwarový systém pro automatizaci a integraci testovacích strojů
Software System for Tester Machine Automation and Integration

Jazyk vypracování: čeština

Zásady pro vypracování:

Cílem práce je implementace softwarového systému, který bude automatizovat a integrovat testovací stroje ve společnosti On Semiconductor. Výsledek práce bude implementace obecného řešení, které bude po připojení další softwarové komponenty - driveru, umožňovat budoucí použití pro jakýkoli testovací stroj. K obecnému řešení budou implementovány 2 drivery, které budou použity pro 2 konkrétní testovací stroje.

1. Seznámení se s výrobou a testování čipů v polovodičovém průmyslu.
2. Návrh a implementace obecné části softwarového systému.
3. Seznámení se s testovacím strojem Powertech QT 6100 a ETS 364.
4. Návrh a implementace softwarového driveru pro Powertech QT 6100 a ETS 364.
5. Testování obecného řešení s jednotlivými drivery a konkrétním strojem.
6. Shrnutí dosažených výsledků.

Seznam doporučené odborné literatury:

Podle pokynů vedoucího diplomové práce.

Formální náležitosti a rozsah diplomové práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí diplomové práce: **Ing. Stanislav Martinek**

Konzultant diplomové práce: Ing. Petr Olivka, Ph.D.

Datum zadání: 01.09.2016

Datum odevzdání: 28.04.2017

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

VYSOKÁ ŠKOLA BÁŇSKÁ-TECHNICKÁ UNIVERZITA * Fakulta elektrotechniky a informatiky * 1D16 * OSTRAVA *

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlašuji, že jsem tuto diplomovou práci vypracoval samostatně. Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

Ve Valašském Meziříčí 23.dubna 2017 .....

Souhlasím se zveřejněním této diplomové práce dle požadavků čl. 26, odst. 9 Studijního a zkušebního řádu pro studium v magisterských programech VŠB-TU Ostrava.

Ve Valašském Meziříčí, 20.dubna 2017

.....

Ing. Stanislav Martinek

Rád bych na tomto místě poděkoval vedoucímu diplomové práce, Ing. Stanislavovi Martinkovi, za odbornou pomoc, cenné rady a připomínky při zpracování diplomové práce.

Abstrakt

Diplomová práce se zabývá problematikou integrace a automatizace testovacích strojů ve společnosti On Semiconductor. Na základě analýzy problematiky je navrhnut a implementován softwarový systém, který tento problém obecně řeší.

Klíčová slova: Návrh software, integrace, automatizace, testovací stroj, polovodičový průmysl, testování čipů

Abstract

The diploma thesis focuses on the automation and integration of the tester machines in On Semiconductor. A software system which solves this issue is designed and implemented based on input analysis.

Key Words: Software design, integration, automation, tester machine, semiconductor industry, final test

Obsah

Seznam použitých zkratek a symbolů

FT	–	Final Test
PT	–	Probe Test
UML	–	Unified Modeling Language
Si	–	Silicon
MES	–	Manufacturing Execution System
ATE	–	Automatic Test Equipment
DUT	–	Device Under Test
GPIB	–	General Purpose Interface Bus
STDF	–	Standard Test Data Format
POM	–	Project Object Model

Seznam obrázků

Seznam tabulek

Seznam výpisů zdrojového kódu

1 Úvod

Práce se zabývá automatizací a integrací testovacích strojů ve společnosti On Semiconductor. Testovací stroje jsou používány k testovaní kvality čipů na základě řady elektrických vlastností. V rámci diplomové práce je vytvořen softwarový produkt, který bude automatizovat a integrovat testovací stroje ve společnosti.

Potřeba pro vznik nového softwarového řešení pro testovací stroje byla podnícena dalším krokem v automatizaci, který byl zahájen projektem pro unifikaci MES systémů napříč skupinou On Semiconductor. V rámci společnosti bylo nutné vyřešit problém jednotného přístupu k jednotlivým platformám testovacích strojů s cílem jejich vzdáleného monitorování a kontroly pro dosažení větší efektivity v oblasti testování čipů.

V první části diplomové práce je ve stručnosti popsán proces výroby polovodičů. Další část se obecně zabývá problematikou integrace a automatizace testovacího stroje a zmiňuje dvě konkrétní platformy testovacích strojů. Poslední část diplomové práce řeší architekturu a implementaci softwarového řešení.

1.1 Cíl práce

Cílem práce je implementace softwarového systému, který bude automatizovat a integrovat testovací stroje ve společnosti On Semiconductor. Výsledek práce bude implementace obecného řešení, které bude po připojení další softwarové komponenty - ovladače, umožňovat budoucí použití pro jakýkoli testovací stroj. K obecnému řešení budou implementovány dva ovladače, které budou použity pro dva konkrétní testovací stroje.

2 Oblast výroby polovodičů

V této kapitole jsou popsány jednotlivé kroky při výrobě polovodičů tak, aby byl jasný širší kontext prostředí, do které je výstup diplomové práce zamířen. Je zde popsán základní prvek pro výrobu polovodičů - křemík, způsob výroby křemíkových desek, výroba čipů na křemíkové desky a dvě oblasti testování čipů. Výstup diplomové práce míří oblasti testování kvality zapouzdřených čipů.

Mezi významné výrobní procesy společnosti On Semiconductor patří výroba křemíku pro výrobu čipů. Křemík se s dlouholetou tradicí vyrábí v Rožnově pod Radhoštěm, kde je tento kov následně i zpracováván do podoby křemíkových desek, na které jsou poté v této lokalitě vyráběny i samotné čipy.

V Rožnově pod Radhoštěm se také nachází globální IT centrum celé korporace, v rámci které funguje i ERDC - „European Regional Development Center“, což je skupina týmů složených z analytiků a vývojářů, kteří zajišťují vývoj aplikací pro podporu výroby v celé skupině On Semiconductor.

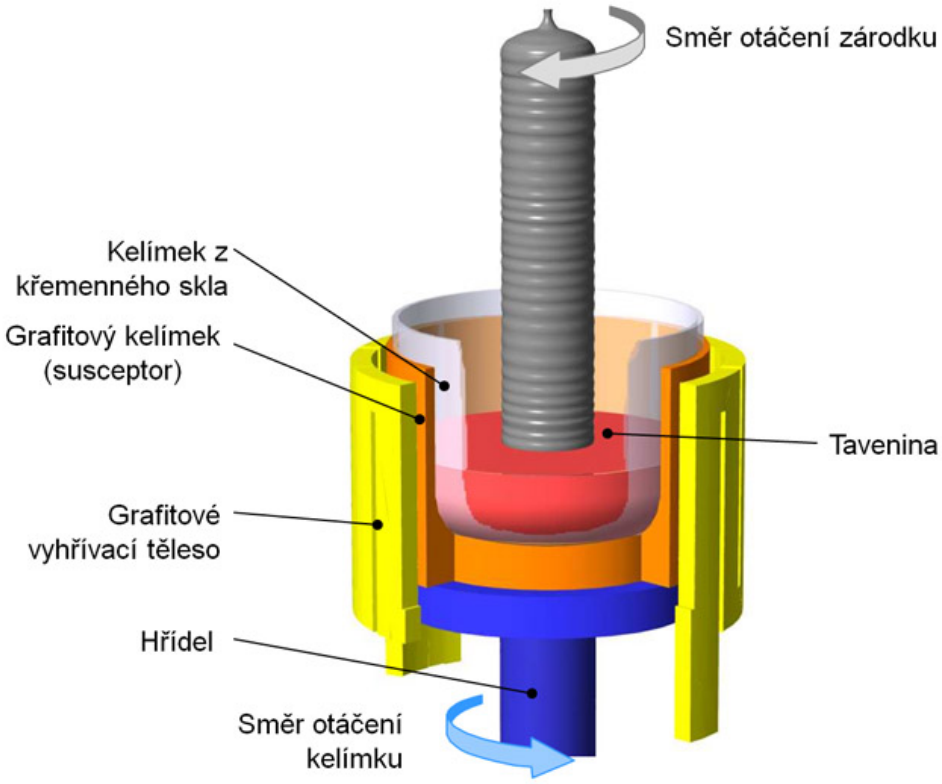
Při popisů jednotlivých výrobních fází bylo vycházeno z [1] a interních materiálů společnosti.

2.1 Zpracování křemíku

Na počátku čipu je prvek zvaný křemík. Za určitých podmínek vede elektrický proud. Křemík je obsažen v hojném množství v zemské kůře. Vyskytuje se zde ve sloučeninách, zejména v křemičitanech nebo křemenu. Pro možnost použití křemíku v polovodičovém průmyslu je nutné křemík izolovat z těchto sloučenin. Pro tento účel se používá křemenný písek, jenž se dalšími procesy redukuje na vysoce čistý křemík - polykrystal - ten však není zcela vhodný pro použití v polovodičovém průmyslu.

Atomy křemíku musí být uspořádány do krystalové mřížky, tj. do monokrystalické formy, která je základem pro výrobu křemíkových desek. Vodivost struktury je poté lehce modifikovatelná přidáním dalších prvků. Křemík dopovaný bórem zaručuje vodivost typu P - přenos elektronů je uskutečněn chybějícím elektronem ve valenční vrstvě atomu křemíku. Křemík dopovaný fosforem má vodivost typu N, kde je náboj přenášen elektrony.

Nejčastěji používaným způsobem výroby monokrystalu křemíku je růst taveniny tzv. Czochralského metoda. Vsádka polykrastalického křemíku v kelímku křemenného skla je vsazena do horní části zařízení na výrobu monokrystalu tzv. tažičky, poté je zařízení uzavřeno a proběhne odsání vzduchu. Proces tažení probíhá za sníženého tlaku v ochranné atmosféře argonu. Po roztavení vsádky polykrystalického křemíku se do taveniny přidá monokrystalický zárodek. Na tažičce jsou poté nakonfigurovány otáčky, posuv zárodku a teplota. Tyto parametry určí průměr konečného monokrystalu. Zárodek je poté táhnout směrem nahoru (viz obrázek 1) a vznikne křemíkový monokrystalický ingot (viz obrázek 2), který je základ pro výrobu křemíkových desek. Cílem je vytvořit monokrystalický ingot s největším možným průměrem tak, aby se zde vešlo maximální možné množství čipů. Nejčastěji se dnes procesory a grafické čipy vyrábí z monokrystalů



Obrázek 1: Tažení monokrystalu[1]

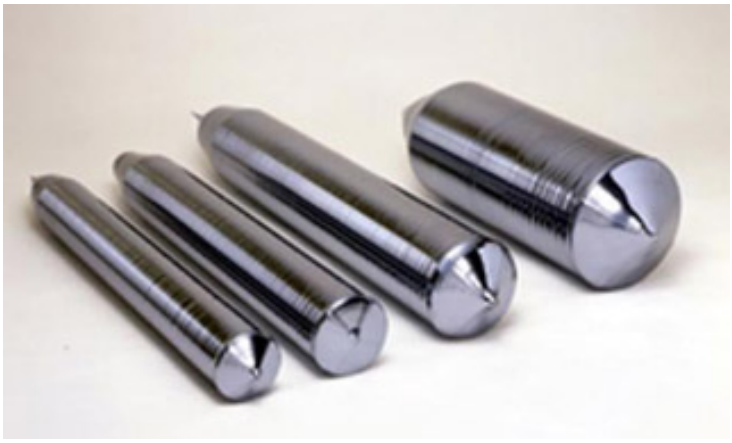
o průměru 300 mm. Technologický pokrok se však nezastavil a výrobce Intel již začal používat monokrystaly o průměru 450 mm.

2.2 Výroba křemíkových desek

Konce monokrystalického křemíkového ingotu jsou odstraněny a ze zbývající části je nařezáno několik testovacích desek, u kterých je měřen měrný elektrický odpor, koncentrace kyslíku a uhlíku. Jedná se pouze o vzorek z celku tak, aby bylo možné odhadnout průběh vlastností u celé části ingotu. Pokud vlastnosti vyhovují, ingot je vybroušen do požadovaného průměru. Desky s tloušťkou 0,3 - 0,7 mm jsou poté řezány buď diamantovou pilou nebo na drátových řezačkách, kde se zpracovává celá část ingotu najednou. Hrany vyřezaných desek jsou následně broušeny diamantovým kotoučem do oblého tvaru tak, aby dosáhly vetší odolnosti proti odlamování v dalším procesu výroby.

Dalším krokem ve výrobě je oboustranné čištění a následné broušení, tzv. lapování. Desky se umístí do držáku, kde jsou broušeny mezi dvěma litinovými lapovacími kotouči. Obě strany desky jsou broušeny současně. Cílem je odstranit z desek část narušeného křemíku z předchozí fáze řezání a současně zarovnat povrch desek.

[1]Zdroj: Interní materiály On Semiconductor



Obrázek 2: Ingot křemíku různých průměrů[2]

Lapování odstraní větší části porušeného křemíku a zaručí jednotnou vrstvu povrchu křemíku. Problém, který nastává je, že vrstva je stále porušena, a tudíž je nutné použít dalších metod bez toho, aniž by na deskách byl zanechán porušený povrch. K tomuto účelu se používá chemické leptání ve směsi kyselin odstraňující zbytkové narušení po lapování.

Dalším krokem je leštění křemíkových desek z jedné strany. Tento chemicko-mechanický proces zaručí hladký, rovný povrch bez jakéhokoli porušení. Přední strana desky je po tomto kroku zrcadlově čistá. Kontrola parametrů desek probíhá bezkontaktní metodou. Měří se elektrický měrný odpor a geometrické parametry. Část leštěných desek je po vizuální kontrole zabaleno a distribuováno přímo k zákazníkům. Druhá část desek pokračuje na fázi epitaxe.

Pro některé typy polovodičových prvků je nutné nanést další vrstvu monokrystalu, která má jiné elektrické vlastnosti než základní deska. Jedná se o velmi tenkou vrstvu obsahující odlišnou koncentraci dopantů od základní vrstvy, avšak její krystalografické vlastnosti jsou stejné. Je zde také možnost použití jiného dopantu. Tato vrstva se vytváří chemickou depozicí křemíku z plynné fáze, tzv. epitaxního růstu.

Leštěné křemíkové desky a desky s epitaxní vrstvou jsou hlavním produktem z monokrystalů křemíku a slouží jako základní materiál pro výrobu polovodičových součástek. Hotové křemíkové desky se balí do přepravních zásobníků - typicky po 25 kusech.

2.3 Výroba čipů

Výroba čipů probíhá na leštěné křemíkové desky, jejichž proces výroby byl popsán v předchozí podkapitole. V této podkapitole budou stručně popsány jednotlivé procesy, které předchází hotovému čipu. Tyto procesy nemusí následovat nutně za sebou, ale mohou se mnohokrát v různém pořadí opakovat tak, aby čip splňoval požadované vlastnosti. Jak již bylo zmíněno dříve, jedna z klíčových vlastností polovodiče je typ vodivosti. Typ vodivosti je možné upravit přidáním vhodného dopantu. Jednotlivé vrstvy, které jsou nanášeny na čistou křemíkovou vrstvu mohou

[2]Zdroj: http://metalsolution.co/images/Siliconingots.jpg

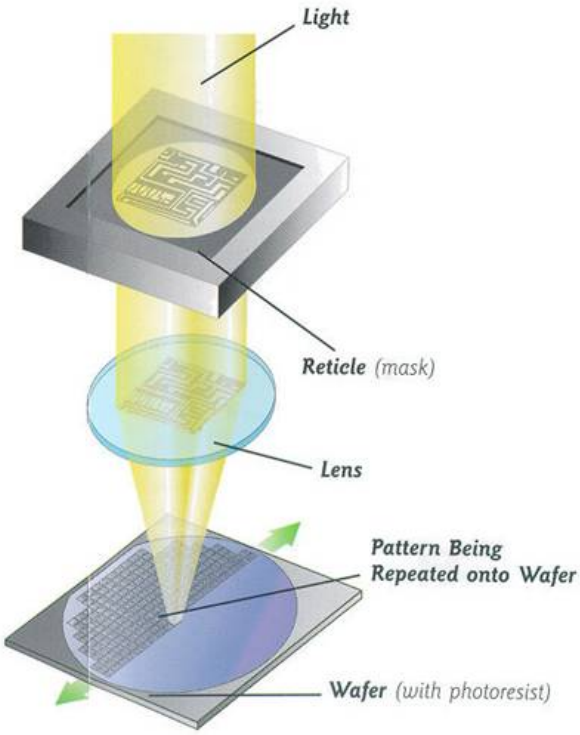
Obrázek 3: Křemíkové desky vkládány do oxidační pece[3]

mít různé typy vodivosti. Kombinace těchto vrstev zaručí požadované vlastnosti výsledného integrovaného obvodu. Jedním z těchto procesů je oxidace. Oxidace se používá k zapouzdření oblasti na křemíkové desce nebo vrstvě, která má být chráněna před nánosem jiné vrstvy. Oxidace je jev, kdy povrch křemíkové desky reaguje s kyslíkem a tvoří zde oxid křemičitý. Tento jev za přítomnosti vzduchu probíhá neustále. Pro využití ve výrobě je nutné jej urychlit pomocí oxidační pece. Pec je rozehřátá na teplotu cca 1000 °C a na povrch křemíkové desky je je puštěn buď čistý kyslík nebo vodní pára. V závislosti na typu integrovaného obvodu se tloušťka oxidační vrstvy může lišit. Řádově však dosahuje tloušťky desítek až stovek nanometrů. Rychlost růstu oxidové vrstvy na povrchu křemíkové desky je závislý na teplotě v oxidační peci, typu oxidantu (kyslík nebo vodní pára) a v neposlední řadě také na krystalografickém uspořádání krystalové mřížky křemíku. Kvůli neustále oxidaci je nutné po nanesení oxidové vrstvy začít co nejdříve s dalším procesem - nejčastěji epitaxe.

Proces, kdy se nechá povrch křemíkové desky růst, se nazývá epitaxe. Tímto procesem je možné nanést na křemíkovou desku další vrstvu křemíkového monokrystalu. Tato vrstva může mít libovolný typ vodivosti. Nejčastěji se tento proces provádí v epitaxním reaktoru, kde teplota dosahuje cca 1000 - 1400 °C. Do epitaxního reaktoru se umístí křemíkové desky, které se nechají zreagovat s plyny, které obsahují atomy křemíku. Pro tento účel bývá používáno chloridu křemičitého nebo silanu. Jedná se o tzv. plynnou epitaxi. Tak jako proces oxidace je epitaxe závislá na teplotě v reaktoru. Běžně je dosahováno růstu přibližně 1 mikrometr za minutu.

Existují dva způsoby jak měnit množství dopantů v křemíku a tím jejich typ vodivost. První

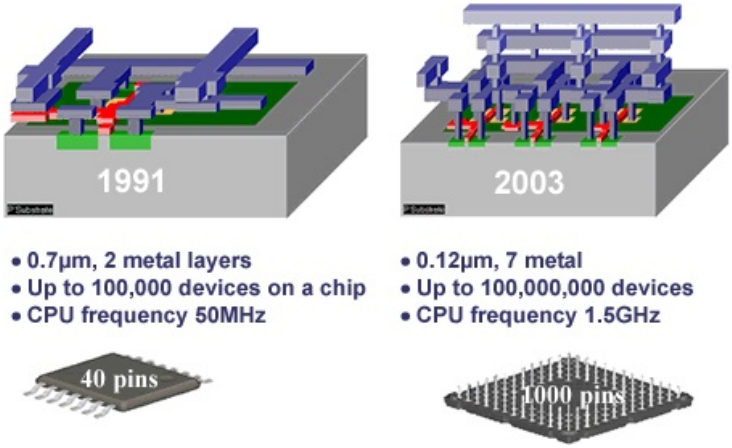
[3]Zdroj: https://www.cismst.org/uploads/pics/fachbereiche_wafer_01.jpg



Obrázek 4: Fotolitografie[4]

způsob je pomocí difúze, kdy se příměsi dostávají na povrch křemíkové desky, a poté se jejich koncentrace rozprostře po určené ploše. Druhý způsob je iontová implantace, kdy se dopanty nastřelují na křemíkovou desku. Difúzí lze upravovat více křemíkových desek současně, ale není zde možnost přidávat malé dávky příměsí tak přesně jak u iontové implantace. Při dopování křemíkové desky se využívá procesu fotolitografie. Fotolitografie je soubor technologických operací umožnující přesnos motivu z předlohy masky na desku. Postup za sebou následujících technologických kroků bude ve stručnosti popsán. První je nutné křemíkovou desku chemicky ošetřit tak, aby k ní dobře přilnul fotorezist. Fotorezist je látka, která při vystavení světlu o určité vlnové délce mění své vlastnosti. Pozitivní fotorezist se vytvrdí. Negativní fotorezist se rozpadne. Po nanesení fotorezistu na desku se deska umístí na plotnu o teplotě cca 100 °C a fotorezist se nechá vytvrdnout. Následuje expozice, kdy je deska umístěna a seřízena vůči masce s motivem a poté maskou prochází UV lasery s vlnovou délkou 248 a 193 nm. Následuje druhé umístění desky do plotny o teplotě cca 120 - 140 °C. Po druhém vytvrzení se deska umístí do chemického roztoku, který nebyl vytvrzen během expozice (negativní fotorezist), případně se během expozice vlivem světla uvolnil (pozitivní fotorezist). Následně je deska usušena a probíhá vizuální kontrola Pokud je vše v pořádku, následuje proces výroby, kvůli kterému byla litografie prováděna. Jak již bylo zmíněno, může se jednat o nanášení oxidu v místech, kde deska není zakryta fotorezistem. V současné době je kladen důraz na co nejmenší možné masky tak, aby se na desku vlezlo co

[4]Zdroj: http://maltiel-consulting.com/How_to_Make_a_Semiconductor_Chip.htm



Obrázek 5: Vývoj složitosti vodivé sítě[5]

nejvíce čipů. Jedna deska může obsahovat tisíce čipů.

Základním předpokladem pro vytvoření kontaktů na integrovaných obvodech je vytvoření vnitřní vodivé sítě. Metalizace je nanášena naprašováním hliníku nebo dnes už běžnější mědi na přední stranu desky. Pro některé typy obvodů se také naprašuje metalická trojvrstva na zadní stranu desky. Moderní integrované obvody jsou složeny z mnoho vrstev kovových spojení. Nejsložitější obvody jich mají až několik desítek. Při vývoji platí pravidlo, že všechny vodiče v jedné vrstvě musí být rovnoběžné a otočené o 90 stupňů z důvodu správného postupu a spolehlivé výroby. Jednotlivé vrstvy jsou propojeny svislými kontakty. Historický vývoj je zobrazen na obrázku č. 5. Vodivostní síť dnešních nejsložitějších obvodů by znázornit už tak jednoduché nebylo.

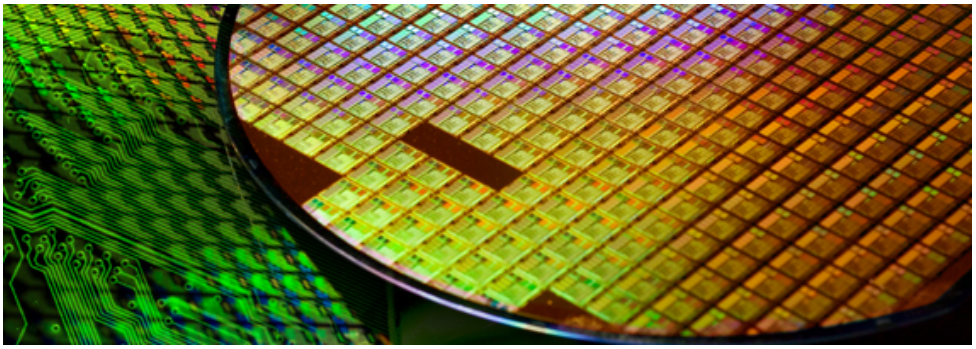
2.4 Testování

Testování kvality materiálu probíhá v průběhu všech fází výroby křemíku. Nejdůležitější testování však přichází ve chvíli, kdy čip je již jasně definován a vytvořen na křemíkovém podkladu. V polovodičovém průmyslu existují dvě oblasti testování čipů. Jedná se o tzv. „**probe test**“ - testování čipů nacházející se na křemíkových deskách a „**final test**“ - testování rozřezaných a zapouzdřených čipů. V následujícím textu budou používány tyto dva pojmy pro referenci na zmiňované oblasti testování. Řešení této práce míří primárně do oblasti „final test”, potenciální využití však bude možné i v „probe testu”.

2.4.1 Testování desek

Testování desek neboli „probe test” se provádí ve chvíli, kdy na křemíkové desce jsou již vytvořeny hotové struktury jednotlivých čipů (tzn. po dokončení poslední vrstvy metalizace). Desku

[5]Zdroj: https://moodle.insa-toulouse.fr/pluginfile.php/2676/mod_resource/content/0/content/images/device_roadmap.gif



Obrázek 6: Jednotlivé čipy na křemíkové desce[6]

s jednotlivými čipy a jejich strukturou je možné vidět na obrázku č. 6. V dnešní době není nezvyklé, že jedna deska obsahuje i čtvrt milionu čipů. Tyto čipy však nedosahují složitosti např. moderních procesorů, ale jsou to o několik řádu jednodušší integrované obvody. Tato oblast testování se označuje jako "probe test" nebo méně časti jako "wafer test". Je zde využíváno stejných platforem testovacích strojů jako ve „final testu", odlišnosti jsou v konfiguraci (hardwarové i softwarové).

Testování se provádí kombinací testovacích strojů a krokovacích strojů. Na držák krokovacího stroje je připojena karta, tzv. "probe card" obsahující integrovaný plošný spoj, ze kterého vedou elektrické vývody - hroty. Tyto hroty propojují testovací stroj a čip na desce. Karta tvoří rozhraní mezi čipem na desce a testovacím strojem. Pro každý typ čipu existuje odpovídající karta. Jak již z popisu vyplývá, tato karta je v krokovacím stroji měnitelná. Krokovací stroj zajišťuje přesun karty a připojení karty k čipu na zadané pozici na desce, poté testovací stroj testuje elektrické parametry čipu.

Jednotlivé parametry testů jsou pečlivě zaznamenávány do tzv. „wafer map", které jsou základním výstupem „wafer probe" testování a vstupem do dalších výrobních procesů. Wafer mapy obsahují souřadnicový systém desky, hodnoty o výsledcích testů čipů a další informace o čipech a desce. Wafer mapy slouží jako základní vstup do analytických nástrojů, kterými je pak možné na základě statistických metod vyhodnotit řadu informací, které mohou výtěžnost zlepšit.

2.4.2 Testování čipů

Jakmile je deska otestována a je změřena kvalita jednotlivých čipů, začne se připravovat na rozřezání na jednotlivé čipy. Deska, aby byla odolná v průběhu předchozích procesů, má ve výstupním stavu z wafer testu tloušťku přibližně 750 µm. Do moderních zařízení se však používají mnohem menší čipy, proto nastupuje proces broušení, který desku zbrousí na cca 50 µm.

Zbroušená deska je poté řezána diamantovou pilou na jednotlivé čipy. Tento proces se skládá ze dvou částí. Nejprve je nutné připevnit desku na adhezivní podložku a až poté rozřezat.

[6]Zdroj: http://www.escomp.com/Blog/wp-content/uploads/2015/02/LargeWafer-300x108.jpg

Diamantová pila se musí při řezání chladit vodou. Mezera mezi jednotlivými čipy nepřesahuje 100 µm.

Jednotlivé rozřezané čipy jsou velice křehké, jejich tloušťka dosahuje cca 50 µm, proto následuje fáze pouzdření, které zvýší odolnost a zajistí odvod odpadního tepla do větší plochy. Pouzdro je tvořeno buď z kovu nebo z keramiky. Nejdůležitějším parametrem pro výběr pouzdra je počet vstupních a výstupních kontaktů, jenž čip obsahuje. Při výběru pouzdra jsou zohledňovány i další parametry např. rychlost čipu a míra odpadního tepla, které čip produkuje.

Jakmile je čip zapouzdřen, přichází na řadu poslední test před tím, než je zabalen a expedován. Zde se již nezkoumá to, zda čip obsahuje vadu, ale testují se možné nastavení čipu a určují se zde produktové řady konečného výrobku (jeho možná škálovatelnost). Jedná se o již zmiňovanou oblast ”final test”.

3 Testovací stroj

V této části je testovací stroj obecně popsán v kontextu automatizace možných operací a integrace stroje do systému. Jedná o se o obecnou analýzu testovacího stroje a jeho operací ve výrobě, které mohou být potencionálně automatizovány.

Testovací stroj, obecně označován jako ATE („Automatic Test Equipment"), je v kontextu polovodičového průmysl komplexní testovací zařízení, které testuje elektrické vlastnosti křemíku nebo hotového integrovaného obvodu (nezávisle na tom, zda je čip již zapouzdřen nebo se nachází na křemíkové desce) - obecně je objekt podléhající testu označován jako DUT („Device Under Test"). Cílem tohoto zařízení je rychle zjistit, zda testovací objekt funguje správně a neobsahuje defekt, případně parametry za jakých testovací objekt ještě funguje správně. Hraniční hodnoty měřených parametrů a jejich závislost na fungování čipu pak od sebe mohou odlišovat produktové řady výrobku.

Cena moderního testovacího stroje je v řádech milionů korun. Obecně lze říci, že cena veškerého vybavení pro polovodičový průmysl se pohybuje ve vyšších číslech. Například cena jednoho propojovacího GPIB kabelu, používaného v produkčním prostředí, se může vyšplhat až do desítek tisíc korun.

Testovací stroj je připojen k PC, které ovládá průběh měření. Dále toto PC bude označováno jako „tester PC". Tester PC obsahuje nainstalovaný měřící software od výrobce, který pomocí test programu řídí samotný proces testování. Test program (psaný např. v programovacím jazyce C), je schopen definovat průběh testování a elektrické parametry testů. Každá změna v test programu je pečlivě evidována a před nasazením do produkčního prostředí je nutné projít řadou verifikačních procesů.

Testovací stroj využívá zařízení, které s testovanými objekty manipuluje. Toto zařízení se ve „final testu" označuje jako „handler", v prostředí „probe testu" se jedná o krokovač („prober"). Testovací stroj s „handlerem" tedy testuje jednotlivé, již zapouzdřené čipy. Testovací stroj s připojeným krokovačem testuje čipy na křemíkových deskách.

3.1 Automatizace

Maximální možná míra automatizace je cílem společností nejen v polovodičovém průmyslu. Automatizace v kontextu testování čipů, což je jedním z cílů této práce, je strojové provedení úkonů nahrazující lidský zdroj. Příkladem je zadávání údajů o měřeném čipu, monitorování měření, validace měřených dat, zamknutí ovládacího panelu atd.

Míra automatizace je vždy závislá na platformě a možnostech daného testovacího stroje. Výrobci testovacích strojů většinou poskytují výchozí nástroj pro automatizaci. Běžným výchozím nástrojem pro automatizaci je např. to, že stroj lze přepnout do externího módu, kdy je možné stroj na základě výrobcem daného protokolu ovládat pomocí jednoho z komunikačních rozhraní. Běžně se využívá komunikačního rozhraní GPIB, RS232 nebo Ethernetu.

Situace je komplikovanější, když je potřeba automatizovat starší testovací stroj, kde nejsou žádné nástroje pro automatizaci nebo je nutné automatizovat operace, které nejsou podporovány. V těchto případech se řeší situace zcela individuálně. V mnoha případech je potřeba situaci řešit s přímo s výrobcem stroje a domluvit případné modifikace nástrojů pro automatizaci. Tímto způsobem není možné řešit případy, kdy výrobce stroje již neexistuje, a není tak z jeho strany zajištěna žádná podpora.

Diplomová práce se zabývá automatizací dvou testovacích strojů - testovacím strojem řady QT 6000 od firmy Powertech a testovacím strojem řady ETS 300 od firmy Eagle. Tyto firmy stále existují, je tak možné využít zákaznické podpory. Toho bylo využito u testovacího stroje od Powertechu, jehož výrobce rozšířil komunikační protokol a implementoval novou funkčnost, která byla požadována. Eagle využívá podpory automatizace v možnosti implementování vlastní automatizace uvnitř test programu. Toho však u ETS 364 není využito a využívá se proprietární řešení FWMC. Automatizace jednotlivých platforem je popsána v příslušných podkapitolách.

V následujících částech je obecný výpis operací prováděných s testovacím strojem a popis možných automatizací v rámci jednotlivých operací. Operace mohou, ale nemusí mít podporu u konkrétní platformy testovacího stoje.

3.1.1 Start testování lotu

Start testování lotu je operace, kdy na vstupu stroje je připraven lot (sada čipů) k testování. Operace start testování lotu vždy zahrnuje inicializaci testovacího stroje a nahrání korektního test programu. Test program je vybrán na základě typu testovaného objektu. Další podoperace pro spuštění lotu jsou již závislé na dané platformě. Pro osobu obsluhující testování tato operace znamená zadání parametrů na vstup měřícímu softwaru. Jedná se o parametry - identifikátor lotu, identifikátor testovacího objektu, název test programu - výčet se může dle platformy lišit, tyto jsou však povinné. Automatizace v rámci této operace:

- **Nastavení požadovaných parametrů** - Nastavení požadovaných identifikátorů v testovacím stroji pro úspěšné spuštění testovaní lotu.
- **Nahrání test programu** - Získání test programu ze vzdáleného repozitáře test programů a následné zavedení do testovacího stroje.
- **Validace test programu** - Validace kontrolního součtu souboru test programu.
- **Smazání dat z předchozího měření** - Smazání dat z měření předchozího lotu.
- **Zamknutí ovládacího panelu** - Zamknutí fyzického ovládacího panelu testovacího stroje, tak aby při testování nedošlo k nevyžádanému vstupu z této externí periferie.

3.1.2 Konec testování lotu

Konec testování lotu je operace, kdy byl otestován poslední čip lotu a testovací stroj je připraven na „vyčištění". V tomto kontextu to znamená odehrání test programu z testovacího stroje, ukončení datalogu a závěrečná validace výstupních dat z měření. Testovací stroj se poté dostane do výchozího stavu a je možné začít testování jiného lotu. Automatizace v rámci této operace:

- **Ukončení aktuálního datalogu** - Ukončení zápisu dat z měření do souboru.
- **Odehrání test programu** - Aktuální test program je odehrán z testovacího stroje.
- **Odeslání datalogu do systému sběru dat** - Odeslání datalogu a souvisejících souborů do systému sběru dat.
- **Validace identifikátoru lotu** - Validuje, zda identifikátor lotu v datalogu je stejný jako identifikátor lotu, pod kterým bylo spuštěno testování.
- **Validace času testování** - Validace časových údajů v datalogu. Musí být v časovém intervalu mezi dobou startu testování lotu a konce testování lotu.
- **Validace identifikátoru testovaného objektu** - Validuje, zda identifikátor testovaného objektu v datalogu je stejný jako identifikátor testovaného objektu, pod kterým bylo spuštěno testování.

3.1.3 Start testování sublotu

Start sublotu je operace, která může následovat po startu testování lotu. Sublotem je označována část lotu. V oblasti „final testu" to znamená určitý počet čipů. V „probe testu" by se ekvivalentně prováděla operace „start testování desky". Automatizace v rámci této operace:

- **Nastavení požadovaných parametrů** - Nastavení požadovaných identifikátorů v testovacím stroji pro úspěšné spuštění testování sublotu.
- **Začátek nového datalogu** - Nastavení testovací stroje tak, aby začal ukládat data o měření do nového souboru.
- **Smazání dat z předchozího měření** - Smazání dat z měření předchozího sublotu.

3.1.4 Konec testování sublotu

Jakmile je otestován poslední čip sublotu, měla by následovat operace konec testování sublotu. V této operaci probíhá ukončení aktuálního datalogu. Test program zůstává nahrán v testovacím stroji. Pokud validace a sběr datalogů probíhá v procedurách „ukončení testování sublotu", již se poté neopakuje v „konec testování lotu". Automatizace v rámci této operace:

- **Ukončení aktuálního datalogu** - Ukončení zápisu dat z měření do souboru.

- **Odeslání datalogu do systému sběru dat** - Odeslání datalogu a souvisejících souborů do systému sběru dat.
- **Validace identifikátoru lotu** - Validuje, zda identifikátor lotu v datalogu je stejný jako identifikátor lotu, pod kterým bylo spuštěno testování.
- **Validace času testování** - Validace časových údajů v datalogu. Musí být v časovém intervalu mezi dobou startu testování sublotu a konce testování sublotu.
- **Validace identifikátoru testovaného objektu** - Validuje, zda identifikátor testovaného objektu v datalogu je stejný jako identifikátor testovaného objektu, pod kterým bylo spuštěno testování.

3.1.5 Aktuální stav lotu

Aktuální stav lotu je operace, při níž je testovací stroj schopen vrátit aktuální stav testovaného lotu.

3.1.6 Aktuální stav stroje

Aktuální stav stroje je operace, při níž je testovací stroj schopen vrátit identifikátor aktuálně nahraného test programu, verzi firmware, verzi měřícího software, případně další informace o stroji.

3.1.7 Aktuální souhrn měření

Aktuální souhrn měření je operace, při níž je testovací stroj schopen poskytnout informaci o aktuálně změřených čipech v rámci lotu.

3.1.8 Kalibrace stroje

Kalibrace zařízení je operace, díky níž je testovací stroj uveden do kalibračního stavu. Tento úkon ve většině případů nebude automatizován. Automatizace zde může být ve smyslu nastavení vstupních parametrů pro úspěšné uvedení do tohoto stavu. Samotná kalibrace je prováděna manuálně u stroje.

3.1.9 Údržba stroje

Údržba zařízení je operace, která testovací stroj uvede do stavu údržba zařízení. Podobně jako u předchozí operace „kalibrace stroje", údržba probíhá manuálně u stroje.

3.1.10 Monitorování výtěžnosti

Operace monitorování výtěžnosti je kontrola výtěžnosti v periodických časových intervalech.

- **Monitorování** - Monitorování aktuální výtěžnosti a v případě porušení definovaných limitních hodnot je nutné upozornit klienta.

3.1.11 Monitorování testování

Operace monitorování testování je periodická kontrola, zda nedochází k testování bez toho, aniž by o tom věděl klient.

- **Monitorování** - Monitorování spuštění testování na testovacím stroji. V případě, že je spuštěno testování manuálně - mimo správu systému, je nutné upozornit klienta.

3.2 Integrace

Integrace testovacích strojů ve skupině On Semiconductor bude znamenat sjednocení jednotlivých operací, které je možné provádět vzdáleně s testovacím strojem a sjednocení výstupu dat z testování.

3.2.1 Integrace operací

V rámci řešení je definováno softwarového rozhraní, které bude skrývat odlišnosti jednotlivých platforem. Následně, klient (MES systém) přistupující k testovacímu stroji nebude rozlišovat dle platformy, ale uvidí pouze jednotné rozhraní operací testovacího stroje. Jednotlivé platformy budou toto rozhraní implementovat.

3.2.2 Integrace dat z měření

V kontextu testování čipů, binem je označena kategorie výsledku testu čipu. Bin je tedy číslo, které označuje typ chyby vyskytující se v čipu. Ve většině případů se jedná o číslo od 0 do 256, kdy 0 je označován tzv. „dobrý bin", kdy test neobsahuje žádnou chybu. Typ binu může být buď „hard" (HW) nebo „soft" (SW). Hard binem označujeme kategorii chyby a soft binem podkategorii chyby.

Binová data jsou získávána parsováním dat z měření, které testovací stroj produkuje. Ve většině případů se parsuje z binárního souboru typu STDF - „Standard Test Data Format", což je původní proprietární formát zápisu dat z měření od firmy Teradyne, ale postupně se stal standardem pro zaznamenání měřícím dat v polovodičovém průmyslu. Parsování tohoto souboru je netriviální záležitostí, specifikace formátu STDF v současné verzi má přes 100 stran. Formát STDF není podporován u všech testovacích strojů, proto je nutné sjednotit formu dat, která budou odeslána ke klientovi.

Je definován formát XML elementu „bin”;

- **name**: String
- **number**: Integer
- **passfail**: PASS, FAIL
- **type**: HW, SW
- **count**: Integer

Následuje příklad binových dat zapsána v XML.

```
<BINPARETO>
   <BIN>
      <NAME>BIN#1</NAME>
      <NUMBER>1</NUMBER>
      <PASSFAIL>PASS</PASSFAIL>
      <TYPE>HW</TYPE>
      <COUNT>1000</COUNT>
   </BIN>
   <BIN>
      <NAME>BIN#5</NAME>
      <NUMBER>5</NUMBER>
      <PASSFAIL>FAIL</PASSFAIL>
      <TYPE>HW</TYPE>
      <COUNT>10</COUNT>
   </BIN>
   <BIN>
      <NAME>BIN#6 OPENS</NAME>
      <NUMBER>6</NUMBER>
      <PASSFAIL>FAIL</PASSFAIL>
      <TYPE>HW</TYPE>
      <COUNT>140</COUNT>
   </BIN>
   <BIN>
      <NAME>BIN#8</NAME>
      <NUMBER>8</NUMBER>
      <PASSFAIL>FAIL</PASSFAIL>
      <TYPE>SW</TYPE>
      <COUNT>80</COUNT>
```

```
    </BIN>
    <BIN>
        <NAME/>BIN#10</NAME>
        <NUMBER>8</NUMBER>
        <PASSFAIL>FAIL</PASSFAIL>
        <TYPE>SW</TYPE>
        <COUNT>80</COUNT>
    </BIN>
</BINPARETO>
```

Výpis 1: Příklad bin pareta v XML

3.3 Platforma Powertech QT 6100

Powertech je společnost založena v Číně, která je pro On Semiconductor důležitý dodavatel nejen testovacích zařízení, ale také zařízení pro vizuální inspekci a laser markování. Powertech QT 6100 je komplexní testovací stroj se zabudovaným kapacitním testováním stejnosměrným elektrickým proudem. Toto zařízení je možné aplikovat na malé a střední napájecí tranzistory, MOS-FET tranzistory a diody. Zařízení je schopné otestovat až 56 000 testovacích objektů za hodinu. Zařízení tohoto typu je ve skupině On Semiconductor použito v „probe testu" i ve „final testu". Informace o této platformě byly čerpány z oficiálních webových stránek výrobce testovacího stroje [2] a interních materiálů společnosti On Semiconductor.

Obrázek 7: Testovací stroj Powertech QT 6100[7]

3.3.1 Automatizace testovacích strojů Powertech

Komunikace s Powertech QT 6100 probíhá přes komunikační rozhraní ethernet pomocí TCP/IP. S testovacím strojem je od dodavatele dodán měřící software PTSide, kde je možné pro účely automatizace nakonfigurovat číslo portu a adresu socketu, ke kterému se PTSide připojí a naslouchá dle uvedeného protokolu. Měřící software je určen pro operační systém Microsoft Windows XP a vyšší.

3.3.2 Komunikační protokol

V tabulce 1 je pro názornost zobrazená část komunikačního protokolu poskytnutého od výrobce testovacího stroje.

- Formát zprávy je dán následovně:
  „**<typ příkazu>-<parametr1>=<hodnota1>,<parametr2>=<hodnota2>....-MID:<číselný identifikátor zprávy>\n**”
- Každá zpráva obsahuje číselný identifikátor, který se vždy nachází po části s parametry.
- Po typu příkazu vždy následuje znak „**-**”, po kterém následují parametry. Za touto částí je opět znak „**-**”.

[7] Zdroj: http://www.powertechsemi.com/uploads/allimg/150720/1-150H0152428.jpg

- Parametry jsou odděleny znakem „**,**”.
- Nerozlišují se velká a malá písmena.
- V jedné zprávě může být právě jeden příkaz.
- Ukončení zprávy je provedeno „**\n**”

3.3.3 Komunikace s dodavatelem

V průběhu implementační části bylo nutné rozšířit komunikační protokol o novou funkcionalitu. Jednalo se o podpůrný příkaz pro monitorování výtěžnosti. Příkaz, který bylo nutné doplnit, vracel z testovacího stroje aktuální výtěžnost vyjádřenou v procentech, jenž označují poměr úspěšných testů vůči neúspěšným. Komunikace se zástupcem Powertechu byla velice rychlá a doplnění protokolu, včetně implementace požadavků netrvala déle než měsíc. Nový příkaz **„YLD?”**, jehož odpověď je ve formátu „YL-ALL=<yld(nn.nnn)>, S1=<yld>,S2=<yld>”, který obsahuje výtěžnost na všech testovacích „sitách”, byl využit v implementaci ovladače pro Powertech 6100.

3.4 Platforma ETS 364

Společnost Eagle Test Systems, původem z USA, je divize společnosti Teradyne, která tuto společnost zakoupila v roce 2008. ETS 364 je testovací stroj této společnosti. Na rozdíl od již zmíněného Powertechu je ETS 364 zaměřen na širší oblast testování. S tímto testovacím strojem je možné testovat integrované obvody řídící napájení, teplotní čidla, zesilovače, servo kontrolery a produkty pro automobilový průmysl. Stejně jako předchozí platforma, toto zařízení je využito v „probe testu” i ve „final testu”. Informace o této platformě byly čerpány z oficiálních webových stránek výrobce testovacího stroje [3] a interních materiálů společnosti On Semiconductor.

Tabulka 1: Komunikační protokol pro Powertech QT 6100 (vybraná část)

Příkaz	**Vstupní parametry**	**Výstup**	**Popis**
ABRTEST	žádné	OK, ERROR-No test is running, ERROR-Abort failure	Zruší probíhající test.
ACCSLEVEL	-str	OK, ERROR-Not possible to set access level	Nastavení aktuální role: OPERATOR ENGINEER MAINTENANCE.
ACCSLEVEL?	žádné	AC-<OPERATOR, ENGINEER, MAINTENANCE>	Vrátí aktuální roli.
BINCNT?	žádné	BC-bin=quantity,bin=quantity, . . . , ERROR-Program not Loaded	Vrátí počty binů.
DIE?	žádné	DI-nnn, ERROR-Program Not Loaded	Vrátí počet otestovaných čipů.
DLEND	žádné	OK, ERROR-Failed to end datalog	Uzavře STDF soubor.
DLGEN	-path,filename	OK, ERROR-Failed to open file, ERROR-Program not loaded	Nastaví cestu a jméno souboru s informacemi o měření.
DLGEN?	žádné	DG-path,filename,(NOTSTARTED, INPROGRESS, COMPLETED), ERROR-No Data, ERROR–Data generation error	Vrátí aktuální status generování souboru s informacemi o měření.
DLOGI	-n	OK	Nastaví logování každého n-tého čipu.
DLSTART	žádné	OK, ERROR-Failed to start datalog	Start datalogu, pokud je sočasný datalog neuzavřený, je automaticky uzavřen.
EOL	žádné	OK, ERROR-No Data, ERROR–Data generation error	Ukončí lot.
HNDLID	-str	OK,ERROR-Failed to set handler ID	Nastaví identifikátor handleru.
LDBD	-str	OK, ERROR-No Program Loaded	Nastaví identifikátor loadboardu.
LOT	-str	OK, ERROR-No Program Loaded	Nastaví identifikátor lotu.
LOT?	žádné	L-str, ERROR-No Program Loaded	Vrátí identifikátor lotu.
OPER	-str	OK, ERROR-No Program Loaded	Nastaví identifikátor operátora.
OPER?	žádné	OP-str, ERROR-No Program Loaded	Vrátí identifikátor operátora.
PRGN	-str	OK, ERROR-Program Not Found, Load Failed	Nastaví cestu k test programu.
PRGN?	žádné	PR-str, ERROR-No Program Loaded	Vrátí cestu k test programu.
START	žádné	OK, ERROR-No program loaded, Fault	Spustí test program.
VERTST?	žádné	VT-str	Vrátí aktuální verzi měřícího SW.

Obrázek 8: Testovací stroj ETS 364[8]

3.4.1 Automatizace testovacích strojů Eagle

Standardní způsob automatizace, poskytnuta společností Eagle, je v možnosti vložení vlastní DLL knihovny do měřícího software. Zákazníkovi jsou zpřístupněna určitá data a operace testovacího stroje. Zákazník si tak může implementovat vlastní protokol na libovolné transportní vrstvě. DLL knihovna je psána v jazyce C++ a idea spočívá v implementaci zveřejněného softwarového rozhraní. Rozhraní je poté společně při nahrávání test programu zavedeno do vnitřního systému měřícího software „ETS Shell". Bohužel tento typ automatizace se vztahuje pouze pro testovací stroje řady ETS-88, ETS-800 a ETS-2500, nikoliv pro ETS-364. Pro automatizaci bylo vyžadováno použit proprietárního řešení - programu FWMC.

3.4.2 Komunikační protokol

Požadavek pro automatizaci bylo použít program FWMC, který již funguje v produkčním prostředí. Program FWMC simuluje Windows události kliku myši a stisk klávesy a odesílá je do okna měřícího software, který je dodán výrobcem. Komunikace s programem funguje pomocí souborů na lokálním souborovém systému. Program je napsán v programovacím jazyce Visual Basic, je tedy určen pro operační systém Windows. Podporován je operační systém Microsoft Windows XP a vyšší.

[8]Zdroj: http://www.teradyne.com/products/semiconductor-test/ets-364-ets-600-test-systems

Tabulka 2: Komunikační protokol pro FWMC - ETS364 (vybraná část)

Příkaz	**Vstupní parametry**	**Výstup**	**Popis**
STARTLOT	LOTID, SUBLOTID, DEVICE, TESTPRG, OPERATOR, RELOAD_TESTPRG	LOT STARTED, LOT ALREADY STARTED, LOT CANNOT START, FAIL	Start testování lotu.
ENDLOT	LOTID, PRINT_SUMMARY, CONFIRM_PRINT	NO LOT, LOT CANNOT END, FAIL, LOT ENDED	Ukončí testování lotu.
MONITOR	RUNNING_MIN_YIELD_LIMIT, OVERALL_YIELD_UNIT_COUNT, MAX_DELTA_YIELD_SITES, MAX_CONSECUTIVE_FAILURE, RUNNING_YIELD_UNIT_COUNT	žádný	Nastaví parametry monitoru.
MONITOR_COMPONENT_FLAG	FLAG_MONITOR_OVERALL_YIELD, FLAG_MONITOR_SITE_YIELD, FLAG_MONITOR_DELTA_SITE_YIELD, FLAG_MONITOR_CONSECUTIVE_FAILURE	MONITOR_ON	Zapne monitorování výtěžnosti.
END_MONITOR	žádné	MONITOR_OFF	Vypne monitorování výtěžnosti.
STATUS	žádné	STATUSUPDATE LOT_LOTID	Vrátí identifikátor testovaného lotu.

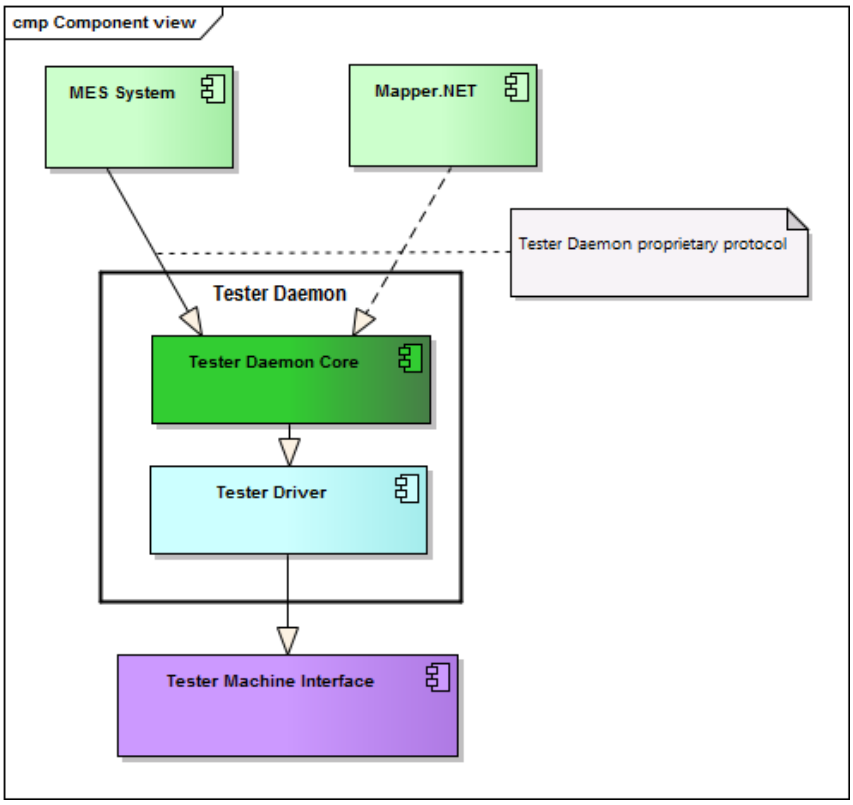
S programem nebyla dodána oficiální dokumentace. Komunikační protokol bylo nutné dekódovat „reverse engineeringem". K dispozici byl zdrojový kód, možnost vidět program fungovat vzdáleným připojením v produkčním prostředí a případná konzultace ze strany automatizačního týmu v Malajsii. Bylo zjištěno, že základem pro komunikaci s programem je konfigurace komunikačních souborů. Pro komunikaci jsou použity dva typy souborů. Jeden soubor pro vstup a druhý soubor pro výstup do programu FWMC. Komunikace probíhá na základě vytvoření a smazání souboru.
Komunikační protokol je popsán následovně:

- Formát zprávy je dán následovně:
  „**COMMAND:<typ příkazu>:~:<klíč>:<hodnota>:~<klíč>:~:<hodnota>..**"
- Data jsou páry skládající se z klíče a hodnoty. Mezi sebou jsou odděleny souborem znaků „**:~:**".
  Příklad: *„LOTID:ON1236.1:~:SUBLOTID:1:~:DEVICE:WT6045"*
- Následuje příklad příkazu, který spustí testování sady čipů nebo desek s označením „ON456T", sublotem nebo deskou „ENG", typem čipu „NCN8026AMNTXG",
  test programem „E36FANCV8853" a operátorem „FG6BGY"
  „COMMAND:STARTLOT:~:LOTID:ON456T:~:SUBLOTID:ENG:~:
  DEVICE:NCN8026AMNTXG:~: TESTPRG:E36FANCV8853:~:OPERATOR:FG6BGY

4 Softwarové řešení

V této kapitole je popsáno jádro celého systému, jehož základem je rozhraní skrývající přístup k různým platformám testovacích strojů, je zde popsán vnitřní stavový stroj, zjednodušeně popsána architektura systému a implementační rysy řešení. V následujících částech bude použito diagramů UML pro modelaci jednotlivých aspektů.

Základní myšlenka softwarového řešení je skryta ve dvou komponentách. První komponenta implementuje obecné řešení a další komponenta implementuje konkrétní komunikační protokol pro daný typ testovacího stroje. V rámci diplomové práce vnikl projekt implementovaný v programovacím jazyce Java s názvem Tester Daemon, který po připojení Tester Driveru tvoří kompletní řešení pro integraci a automatizaci jedné platformy testovacího stroje.



Obrázek 9: Základní komponenty systému

Softwarové řešení Tester Daemon běží typicky na stanici PC, ke které je připojen testovací stroj. Řešení je implementováno primárně pro operační systém Microsoft Windows, ale vzhledem k tomu, že je vyvinuto v Javě, případná migrace, např. na Unix, by neměl být problém. Tester Daemon poskytuje sjednocující vrstvu mezi MES systémem / jiným software, ovládající zařízení, (dále klient) a testovacím strojem. Jeho účelem je skrytí veškerých odlišností mezi jednotlivými platformami testovacích strojů, tzn. poskytnutí jednotného přístupu. To je docíleno definováním množiny operací, které jsou jednotné napříč různými platformami testovacích strojů. Tyto

operace v sobě zahrnují automatizace úkonů popsaných v obecné částí automatizace testovacího stroje.

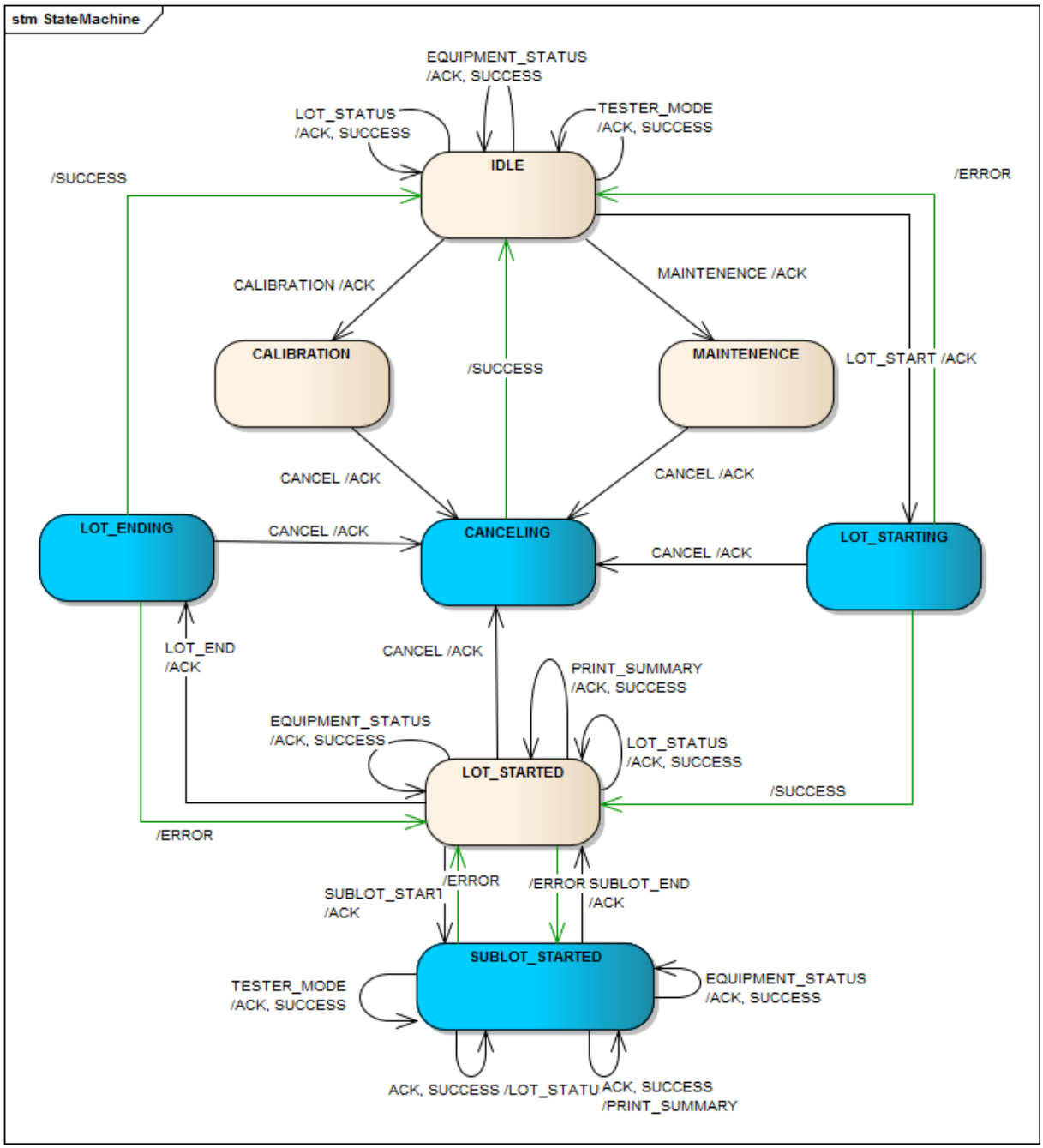
Tester Daemon poskytuje konfigurovatelnou transportní vrstvu. Pomocí této vrstvy je možné s Tester Daemonem komunikovat. Ve výchozím stavu je použito TCP/IP protokolu po ethernetu. Existuje vždy pouze jedna instance komunikace klienta a Tester Daemonu ve stejný čas, avšak je zde možné např. při výpadku spojení znovu připojení a obnovení komunikace ve fázi, kde skončila.

Tester Driver je konfigurovatelná komponenta Tester Daemonu. Tato komponenta je zodpovědná za implementaci jednotlivých operací testovacího stroje.

4.1 Stavy systému

Obecná část systému definuje a implementuje vnitřní chování systému. V rámci systému je definována a implementován stavový automat určující toto chování. Následuje výčet a stručný popis jednotlivých stavů.

- **IDLE** - Klidový stav systému. V tomto stavu je možné provést jakoukoliv operaci se systémem.
- **CALIBRATION** - Stav kalibrace. Stroj je zablokován vůči jakékoliv operaci do doby, než je režim kalibrace ukončen.
- **MAINTENENCE** - Stav údržby. Stroj je zablokován vůči jakékoliv operaci do doby, než je údržba ukončena.
- **CANCELING** - Stav rušení aktuální operace. Tento stav uvede stroj zpět do klidového stavu.
- **LOT_STARTING** - Stav, kdy probíhá startování lotu. V některých případech může trvat i řády minut.
- **LOT_STARTED** - Stav, kdy je lot nastartován.
- **SUBLOT_STARTED** - Stav, kdy je sublot nastartován.
- **LOT_ENDING** - Stav, kdy se ukončuje testování lotu.



Obrázek 10: Stavový stroj systému

V rámci jednotlivých operací jsou definovány přechody mezi jednotlivými stavy. Vnitřní stavový stroj se snaží co nejpřesněji kopírovat stav testovacího stroje. Stavový automat je popsán pomocí diagramu na obrázku č. 10.

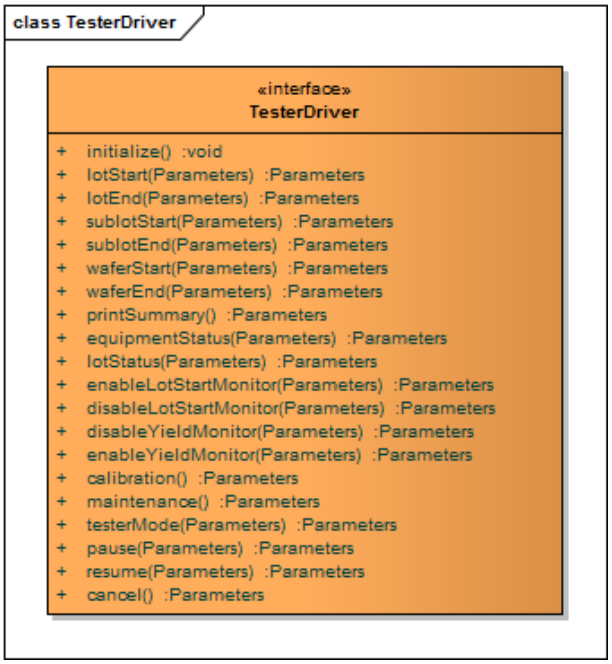
4.2 Obecné rozhraní systému

Obecná část systému definuje rozhraní systému, které je obrazem jednotlivých operací a automatizací pro testovací stroj. Rozhraní je přesně popsáno následujícím diagramem, poté následuje výčet jednotlivých operací rozhraní - jejich vstupů a výstupů.

4.2.1 Operace rozhraní TesterDriver

Jednotlivé operace přijímají vstupní parametry, které jsou ve formě klíč, hodnota. Povinné a nepovinné parametry jsou pro jednotlivé platformy různé. Výstupní parametry jsou ve stejném formátu jako vstupní parametry, tzn. klíč, hodnota. Obecná část nevaliduje vstupy a předává veškeré vstupní data do ovladače, kde jsou následně zpracována.

Jakmile klient odešle požadavek, systém mu odpoví potvrzením („ACK") nebo odmítnutím („ERROR") žádosti s případným důvodem odmítnutí. V případě, že je požadavek přijat, operace je zpracována, a jakmile je dokončena, následuje odeslání zprávy klientovi o zdárném vykonání operace („SUCCESS") nebo chybě ve vykonání operace („ERROR"). Opět, v případě chyby je zde uveden i důvod selhání operace.



Obrázek 11: Rozhraní TesterDriver

- **Start testování lotu**
  - V rozhraní definováno jako „**lotStart**".
  - Základní operace pro začátek testování.

  - Vstupní parametry: identifikátor lotu, identifikátor typu čipu, identifikátor test programu, cesta k test programu, mód test programu, revize test programu, identifikátor handleru, paralelizmus, uzamčení obrazovky
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
  - Stavy systému, ve kterých je operace proveditelná: „IDLE"

- **Konec testování lotu**
  - V rozhraní definováno jako „**lotEnd**".
  - Základní operace pro konec testování lotu.
  - Vstupní parametry: identifikátor lotu, validace identifikátoru lotu (boolean), validace typu čipu (boolean), validace času testování (boolean), požadavek bin pareta (boolean)
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace, bin pareto
  - Stavy systému, ve kterých je operace proveditelná: „LOT_STARTED"

- **Start testování sublotu / desky**
  - V rozhraní definováno jako „**sublotStart**" / „**waferStart**".
  - Základní operace pro začátek testování podmnožiny lotu. Ve „final testu" se používá pojem sublot, naproti tomu v „probe testu" se používá pojem deska.
  - Vstupní parametry: identifikátor lotu, identifikátor typu čipu, identifikátor test programu, cesta k test programu
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace, bin pareto
  - Stavy systému, ve kterých je operace proveditelná: „LOT_STARTED"

- **Konec testování sublotu / desky**
  - V rozhraní definováno jako „**sublotEnd**" / „**waferEnd**".
  - Základní operace pro ukončení testování podmnožiny lotu.
  - Vstupní parametry: identifikátor lotu, validace identifikátoru lotu (boolean), validace typu čipu (boolean), validace času testování (boolean), požadavek bin pareta (boolean)
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace, bin pareto
  - Stavy systému, ve kterých je operace proveditelná: „SUBLOT_STARTED"

- **Získání souhrnu měřených dat**
  - V rozhraní definováno jako „**printSummary**".
  - Operace pro získání souhrnné informace o doposud měřených datech v rámci lotu.
  - Neobsahuje žádné vstupní parametry.
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace, bin pareto, počet celkově otestovaných objektů, výtěžnost
  - Stavy systému, ve kterých je operace proveditelná: „SUBLOT_STARTED", „LOT_STARTED"
- **Získání aktuálního stavu stroje**
  - V rozhraní definováno jako „**equipmentStatus**".
  - Operace pro získání informace o aktuálním stavu zařízení.
  - Neobsahuje žádné vstupní parametry.
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace, identifikátor zařízení, identifikátor měřícího software, mód testovacího stroje
  - Stavy systému, ve kterých je operace proveditelná: „IDLE", „SUBLOT_STARTED", „LOT_STARTED"
- **Získání aktuálního stavu lotu**
  - V rozhraní definováno jako „**lotStatus**".
  - Operace pro získání informace o aktuálním stavu lotu.
  - Neobsahuje žádné vstupní parametry.
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace, identifikátor lotu, identifikátor sublotu, status, identifikátor test programu, cesta k test programu, identifikátor operátora
  - Stavy systému, ve kterých je operace proveditelná: „IDLE", „SUBLOT_STARTED", „LOT_STARTED"
- **Zapnutí monitorování výtěžnosti**
  - V rozhraní definováno jako „**enableYieldMonitor**".
  - Operace pro zapnutí monitorování výtěžnosti.
  - Vstupní parametry: minimální počet otestovaných čipů, časový interval kontroly, limitní hodnota výtěžnosti
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace

  - Bezstavová operace - je možné provést v jakémkoliv stavu.

- **Vypnutí monitorování výtěžnosti**
  - V rozhraní definováno jako „**disableYieldMonitor**".
  - Operace pro vypnutí monitorování výtěžnosti.
  - Neobsahuje žádné vstupní parametry.
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
  - Bezstavová operace - je možné provést v jakémkoliv stavu.

- **Zapnutí monitorování startu testování**
  - V rozhraní definováno jako „**enableLotStartMonitor**".
  - Operace pro zapnutí monitorování startu testování.
  - Vstupní parametry: časový interval kontroly
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
  - Bezstavová operace - je možné provést v jakémkoliv stavu.

- **Vypnutí monitorování startu testování**
  - V rozhraní definováno jako „**disableLotStartMonitor**".
  - Operace pro vypnutí monitorování startu testování.
  - Neobsahuje žádné vstupní parametry.
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
  - Bezstavová operace - je možné provést v jakémkoliv stavu.

- **Režim kalibrace**
  - V rozhraní definováno jako „**calibration**".
  - Operace pro uvedení testovacího stroje do režimu kalibrace.
  - Neobsahuje žádné vstupní parametry.
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
  - Stavy systému, ve kterých je operace proveditelná: „IDLE"

- **Režim údržby**

  - V rozhraní definováno jako „**maintenance**”.
  - Operace pro uvedení testovacího stroje do režimu údržby.
  - Neobsahuje žádné vstupní parametry.
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
  - Stavy systému, ve kterých je operace proveditelná: „IDLE”

- **Změna testovacího módu**
  - V rozhraní definováno jako „**testerMode**”.
  - Operace pro změnu testovacího módu na testovacím stroji.
  - Vstupní parametry: testovací mód
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
  - Stavy systému, ve kterých je operace proveditelná: „IDLE”

- **Operace zrušení**
  - V rozhraní definováno jako „**cancel**”.
  - Operace, která zruší průběh aktuálně vykonávané operace.
  - Neobsahuje žádné vstupní parametry.
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
  - Stavy systému, ve kterých je operace proveditelná: Všechny stavy kromě „IDLE” a „CANCELING”.

- **Operace pauza**
  - V rozhraní definováno jako „**pause**”.
  - Operace, která zastaví testování.
  - Neobsahuje žádné vstupní parametry.
  - Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
  - Stavy systému, ve kterých je operace proveditelná: „SUBLOT_STARTED”, „LOT_STARTED”

- **Operace pokračování**
  - V rozhraní definováno jako „**resume**”.

- Operace, která způsobí pokračování v testování.
- Neobsahuje žádné vstupní parametry.
- Výstupní parametry: standardní odpověď o úspěšném / neúspěšném provedení operace
- Stavy systému, ve kterých je operace proveditelná: „SUBLOT_STARTED", „LOT_STARTED"

4.3 Popis protokolu

Pro komunikace s Tester Daemonem je definován proprietární protokol. Komunikace probíhá buď na základe žádosti a odpovědi nebo je zde možno využít vyvolání události, kdy Tester Daemon iniciuje komunikaci za určitých podmínek. Jako příklad je možné uvést funkcionalitu monitorování, která způsobí, že systém automaticky kontaktuje klienta (MES systém) v případě, že je porušena některá z podmínek monitorů.

4.3.1 Zpráva

V rámci proprietárního protokolu jsou definovány tyto pravidla zpráv:

- Každá zpráva obsahuje číselný identifikátor.
- Více než jeden bílý znak je sloučen do jednoho.
- Nerozlišují se velká a malá písmena.
- V jedné zprávě může být právě jeden příkaz.
- Ukončení zprávy je provedeno „**\n**"

4.3.2 Formát zprávy

V rámci proprietárního protokolu je definován následující formát zpráv:

- Formát zprávy je dán následovně: **MID:<ID zprávy><data><ukončovací znak>**.
- Každá zpráva obsahuje prefix **"MID:"**.
- Data jsou páry skládající se z klíče a hodnoty oddělený dvojtečkou „**:**".
  Příklad: „LOT:ON1236.1"
- Pokud je zde více párů klíč a hodnota, jsou odděleny mezerou.
  Příklad: „LOT:ON1236.1 DEVICE:WT6045"
- Bílé znaky v klíči jsou zakázány.

- Pokud je potřeba mezera v hodnotě, musí být cela hodnota v dvojitých uvozovkách.

  Příklad: „ERROR_MSG:”Something is wrong””

- Následuje příklad příkazu, který startuje testování sady čipů nebo desek s označením „ON1236.1” a typem čipu „WT6045”:

  Příklad: „MID:123 COMMAND:LOT_START LOT:ON1236.1 DEVICE:WT6045\n”

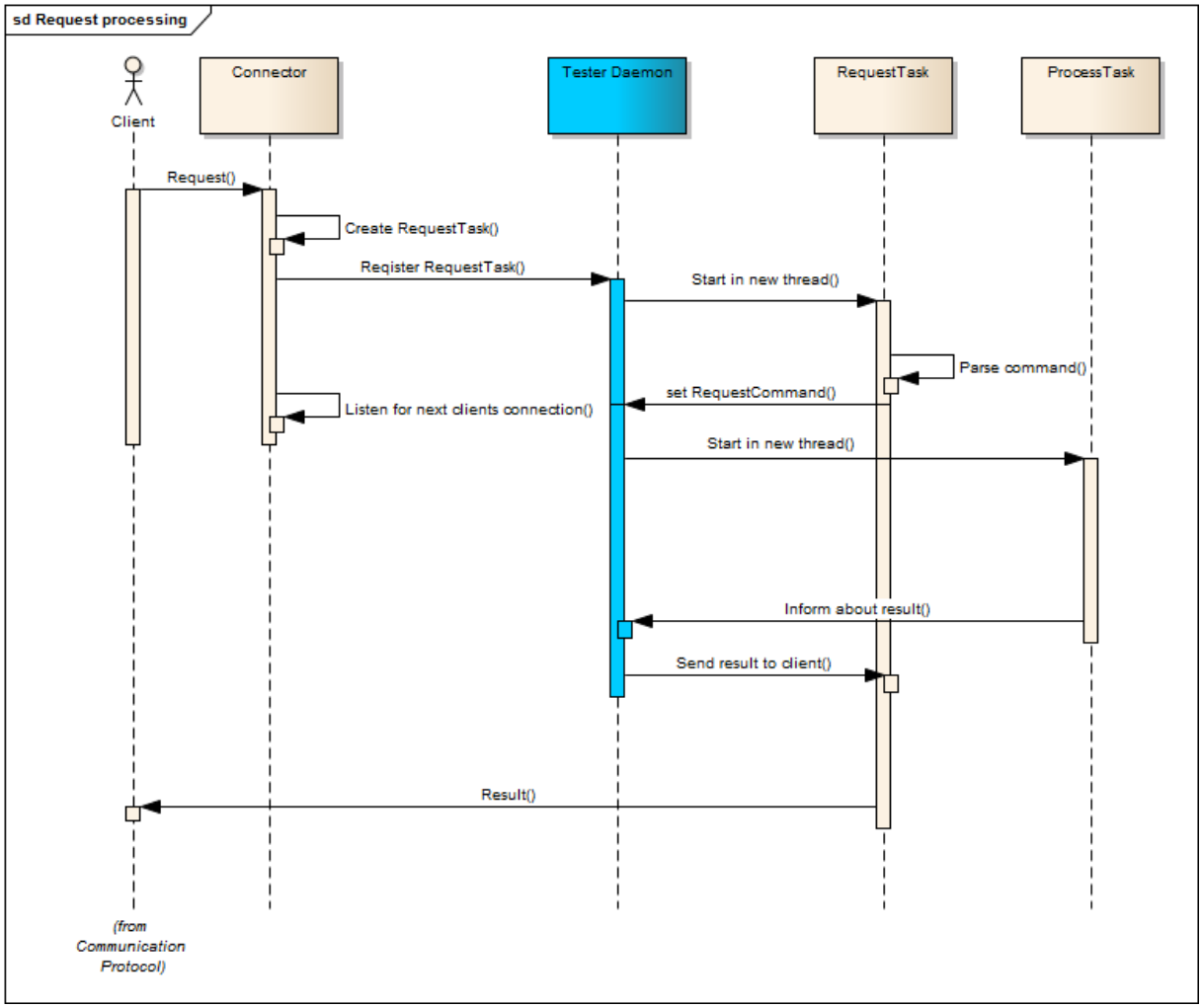
4.3.3 Příklad komunikace

Následuje příklad komunikace po definovaném protokolu, ukázány jsou základní dvě operace - start testování lotu a konec testování lotu.

```
Klient:
MID:123 COMMAND:LOT_START LOT:SBRH12345 DEVICE:0HIST003 OPERATOR:fg6bgy
TEST_PROGRAM_NAME:TP1234fwf23 TEST_PROGRAM_MODE:FT
TEST_PROGRAM_CHECKSUM:6985124578 LOCK_SCREEN:Y

TD:
MID:123 ANSWER:ACK

Jakmile TD úspěšně zpracuje žádost:
MID:123 ANSWER:SUCESS COMMAND:LOT_START


Klient:
MID:356 COMMAND:LOT_END LOT:SBRH12345 DESTINATION_PATH:"\\datacollection\ets300"
UNLOAD_TEST_PROGRAM:Y REQUIRED_FILE_EXTENSIONS:"std,log,xls" VALIDATE_STDF_FILE:Y
VALIDATE_LOTID:Y VALIDATE_TEST_START_TIME:Y TEST_START_TIME_INTERVAL:360
REQUIRE_BIN_PARETO:Y PRINT_SUMMARY:Y LOCK_SCREEN:Y

TD:
MID:356 ANSWER:ACK

Jakmile TD úspěšně zpracuje žádost:
MID:356 ANSWER:SUCCESS COMMAND:LOT_END SUMMARY_RESULT_PRINTED:"N"
BIN_PARETO:"<BINPARETO><BIN><NAME/>BIN#10</NAME><NUMBER>8</NUMBER>
<PASSFAIL>FAIL</PASSFAIL><TYPE>SW</TYPE><COUNT>80</COUNT></BIN>
</BINPARETO>"
```

4.4 Vnitřní architektura

V softwarovém řešení systému je vnitřní proces zpracování požadavků založen na návrhovém vzoru pozorovatel. V systému je definováno několik rozhraní, které jsou využita k tomu, aby se řídící objekt dozvěděl o výsledku operace a mohl na něj reagovat. Následuje sekvenční diagram zachycující zpracování žádosti uvnitř systému a zjednodušený popis celého mechanismu.

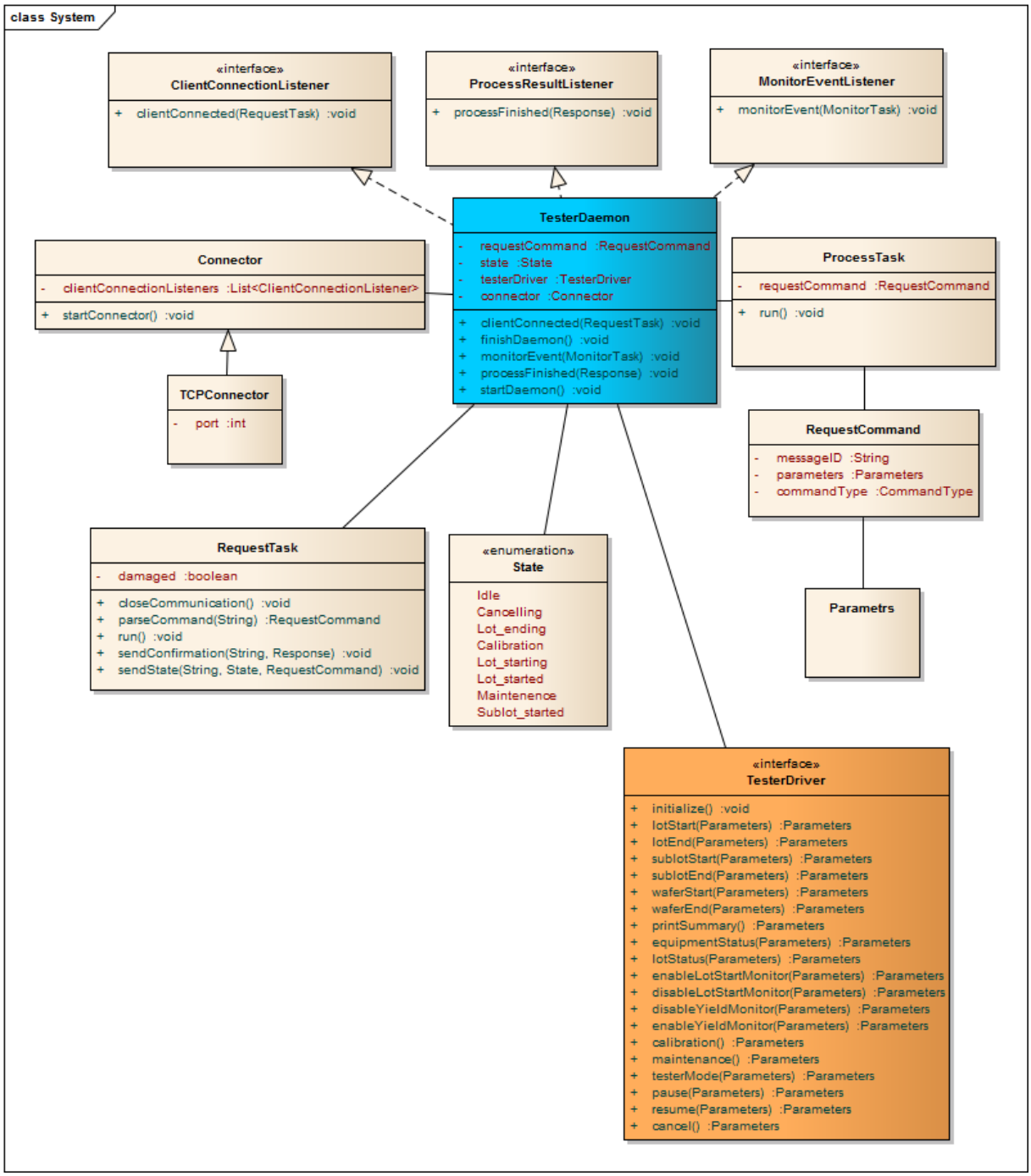


Obrázek 12: Zpracování žádostí uvnitř systému

„Connector" je rozhraní, jehož implementace „TcpIpConnector" je řešení pro TCP/IP transportní vrstvu. Řešení je připraveno pro implementaci jakékoliv jiné transportní vrstvy. Jakmile dorazí zpráva, je předáno řízení do řídící třídy „TesterDaemon", která spustí v novém vlákně dekódování zprávy. „RequestTask" je třída zodpovědná za dekódování zprávy od klienta. V případě chyby dekódování je tato část zodpovědná za upozornění klienta, žádost se tak nedostane do řídící třídy. Pokud je vše v pořádku, je upozorněna řídící třída. Řídící třída poté v novém vlákně spustí vykonání žádosti. „ProcessTask" je třída zodpovědná za vykonání žádosti v dané

implementaci rozhraní „TesterDriver". Komunikace s testovacím strojem je řešena v rámci dané implementace „TesterDriver". Řídící třída je informována o průběhu a zpracování žádosti.

Obecné řešení obsahuje přes 30 tříd - v rámci zachování přehlednosti jsou v diagramu tříd zobrazeny pouze klíčové třídy systému.



Obrázek 13: Přehled klíčových tříd systému

4.5 Ovladače pro testovací stroje

Softwarové ovladače pro testovací stroje jsou v rámci této diplomové práce komponenty systému, které jsou připojeny do obecné části systému. Jednotlivé ovladače implementují obecné rozhraní pro daný testovací stroj. Připojení ovladače k systému probíhá s využitím mechanismu vkládání závislostí („dependency injection"). Tento mechanismus je také využit pro konfiguraci jednotlivých ovladačů pomocí konfiguračního souboru. Více o tomto mechanismu je popsáno v kapitole 5 - „Implementační rysy softwarového řešení".

V konfiguračních souborech u obou ovladačů je možné konfigurovat TCP port, na kterém bude systém naslouchat pro ovládání testovacího stroje. Je zde možné nastavit také dobu čekání v jednotkách sekund na provedení operace, než je zasláno chybové hlášení.

Součástí obou ovladačů je zaznamenání průběhu operací do log souboru. Logují se zde všechny vstupní/výstupní parametry, veškeré data zaslaná nebo přijatá z testovacího stroje, jednotlivé konfigurační položky. V případě, že by nastal problém v produkčním prostředí, bude možné z logovacího souboru rychle zjistit a opravit příčinu vzniku chyby.

Validování výstupních dat z měření je prováděna způsobem popsaným v podkapitole „Automatizace", kapitole „Testovací stroj".

V rámci diplomové práce byly implementovány dva softwarové ovladače. U každého z ovladačů jsou detailněji popsány dvě základní operace - spuštění testování lotu a konec testování lotu. Z důvodu přehlednosti jsou v diagramech znázorněny pouze záležitosti specifické pro daný ovladač a je zde předpoklad úspěšného průběhu. V diagramech nejsou znázorněny všechny chybové průběhy operací, validace parametrů, konfiguračních položek a logování průběhu operací.

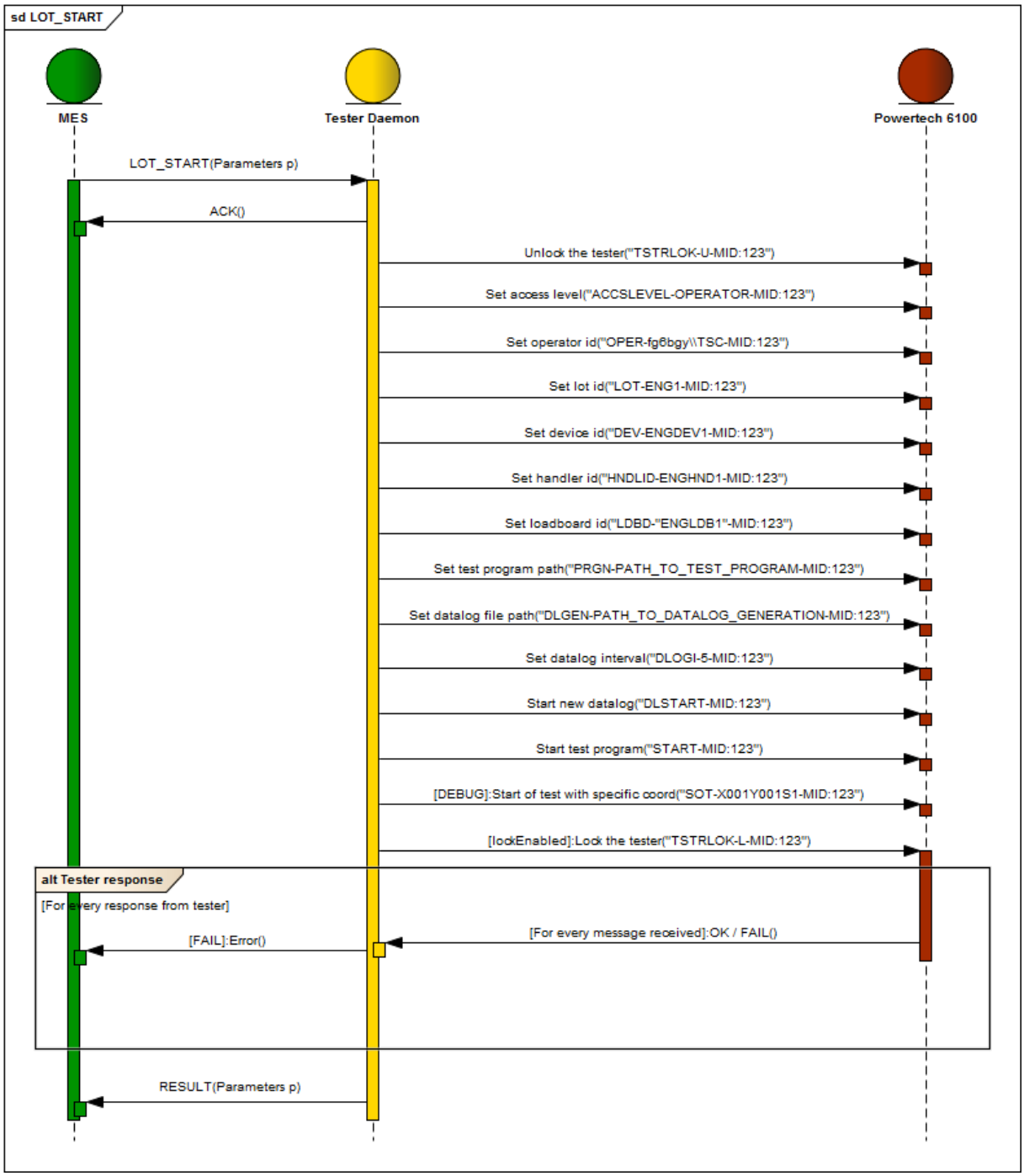
V obou ovladačích se podařilo implementovat následující operace obecného rozhraní:

- **Start testování lotu**
- **Konec testování lotu**
- **Získání souhrnu měřených dat**
- **Získání aktuálního stavu stroje**
- **Získání aktuálního stavu lotu**
- **Změna testovacího módu**
- **Operace zrušení**
- **Zapnutí monitorování startu testování**
- **Vypnutí monitorování startu testování**
- **Zapnutí monitorování výtěžnosti**
- **Vypnutí monitorování výtěžnosti**

4.5.1 Ovladač pro Powertech QT 6100

Ovladač pro Powertech QT 6100 zapouzdřuje komunikaci s testovacím strojem po TCP/IP vrstvě pomocí daného komunikačního protokolu a automatizaci v rámci jednotlivých operací. Konfigurační soubor obsahuje konfiguraci TCP portu pro komunikaci s testovacím strojem.

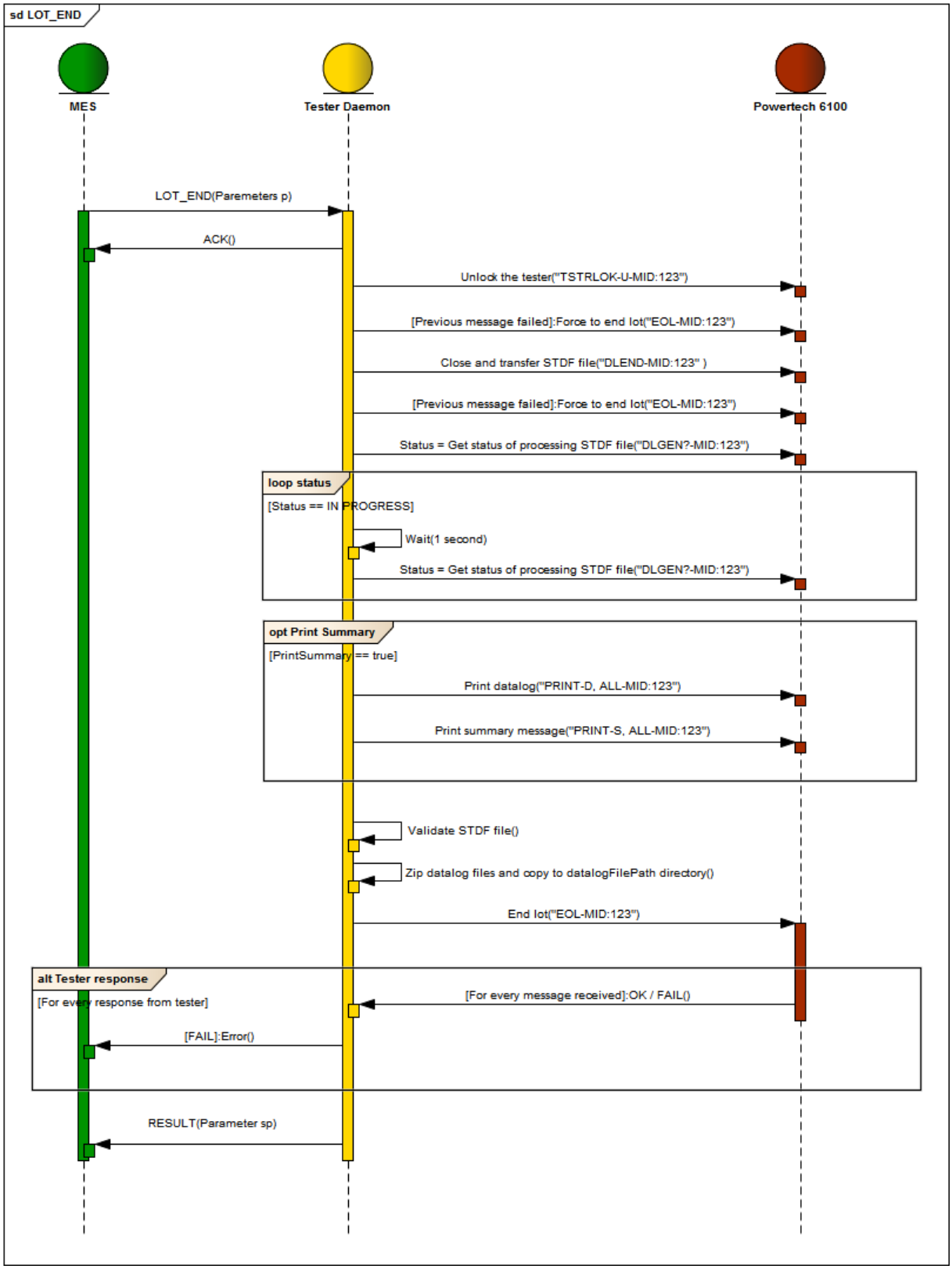
Na začátku operace start testování lotu u Powertech QT 6100 se nejprve stroj odemkne pro vstup. V běžném provozním režimu jsou poté nastavena přístupová práva „operátor", to zajistí přístup pouze k vyhrazeným operacím pro danou roli. Důležitou operací je automatizace vyplnění identifikačních dat vztahujících se k testovanému objektu - jedná se o identifikátory operátora, lotu, testovaného produktu, handleru, loadboardu. Jednou z nejdůležitějších automatizovaných operací při startu měření je automatizace nahrání korektního testovacího programu, který definuje parametry testování. Tento úkon se děje ihned po nastavení požadovaných identifikátorů. Dále je nastavena cesta datalogu. Poslední dvě podoperace v této operaci jsou spuštění test programu a zamknutí testovacího stroje. Jakmile jsou všechny podkroky této operace dokončeny, testovací stroj je ve fázi měření a probíhá testování jednotlivých objektů. Následuje sekvenční diagram, který popisuje přesné pořadí jednotlivých operací v daném komunikačním protokolu.



Obrázek 14: Sekvenční diagram operace start testování lotu na Powertech 6100

Operace konec testování lotu je párová s operací spuštění testování lotu. První je provedeno odemknutí testovacího stroje, pokud se nepodaří odemknout stroj, je provedeno okamžité ukončení lotu a je zasláno chybové hlášení klientovi. V tomto kroku je také provedeno ukončení a zkopírování datalogu do složky, která slouží jako vstup pro systém sběru dat. V případě, že je vše v pořádku a stroj je odemčen, je posílán dotaz v cyklu, zda je již datalog vytvořen. Pokud

ano, je to požadováno, vytiskne se souhrn informací na tiskárně pomocí příkazu „PRINT-D" a „PRINT-S". Dále je provedeno dekódování, validování a vytvoření jednotné formy výstupu dat z měření. Následně je datalog zabalen do archivu zip a zkopírován do vstupu systému pro sběr dat. Následně je testování lotu ukončeno příkazem „EOL". Tento příkaz ukončí běžící test program na testovacím stroji a nastaví stroj do výchozího nastavení.



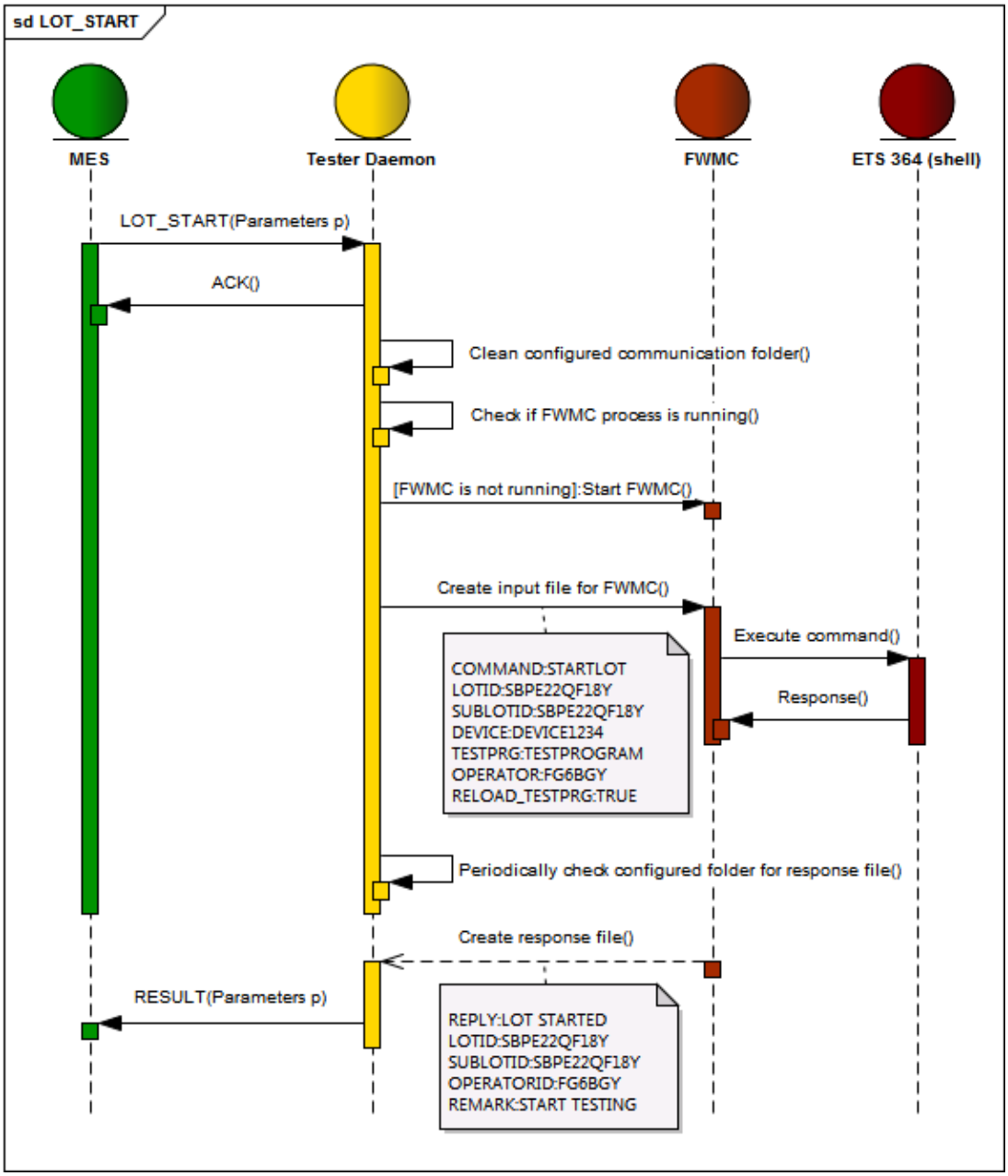
Obrázek 15: Sekvenční diagram operace konec testování lotu na Powertech 6100

4.5.2 Ovladač pro ETS 364

Podobně jako předchozí ovladač i ovladač pro ETS 364 zapouzdřuje komunikaci s testovacím strojem a automatizace v jednotlivých operací. Komunikace s testovacím strojem probíhá přes soubory na lokálním disku. Cesty k těmto souborům jsou součástí konfiguračního souboru pro tento ovladač. Součástí konfiguračního souboru jsou tyto konfigurační položky:

- **Konfigurace cesty k aplikaci FWMC**
- **Konfigurace cesty k vstupním souborům pro komunikaci**
- **Konfigurace cesty k výstupním souborům pro komunikact**
- **Konfigurace cesty k datalogu**
- **Konfigurace přípony souboru s datalogem**
- **Více konfiguračních položek týkající se validování datalogu pomocí logu z ETS Shell**

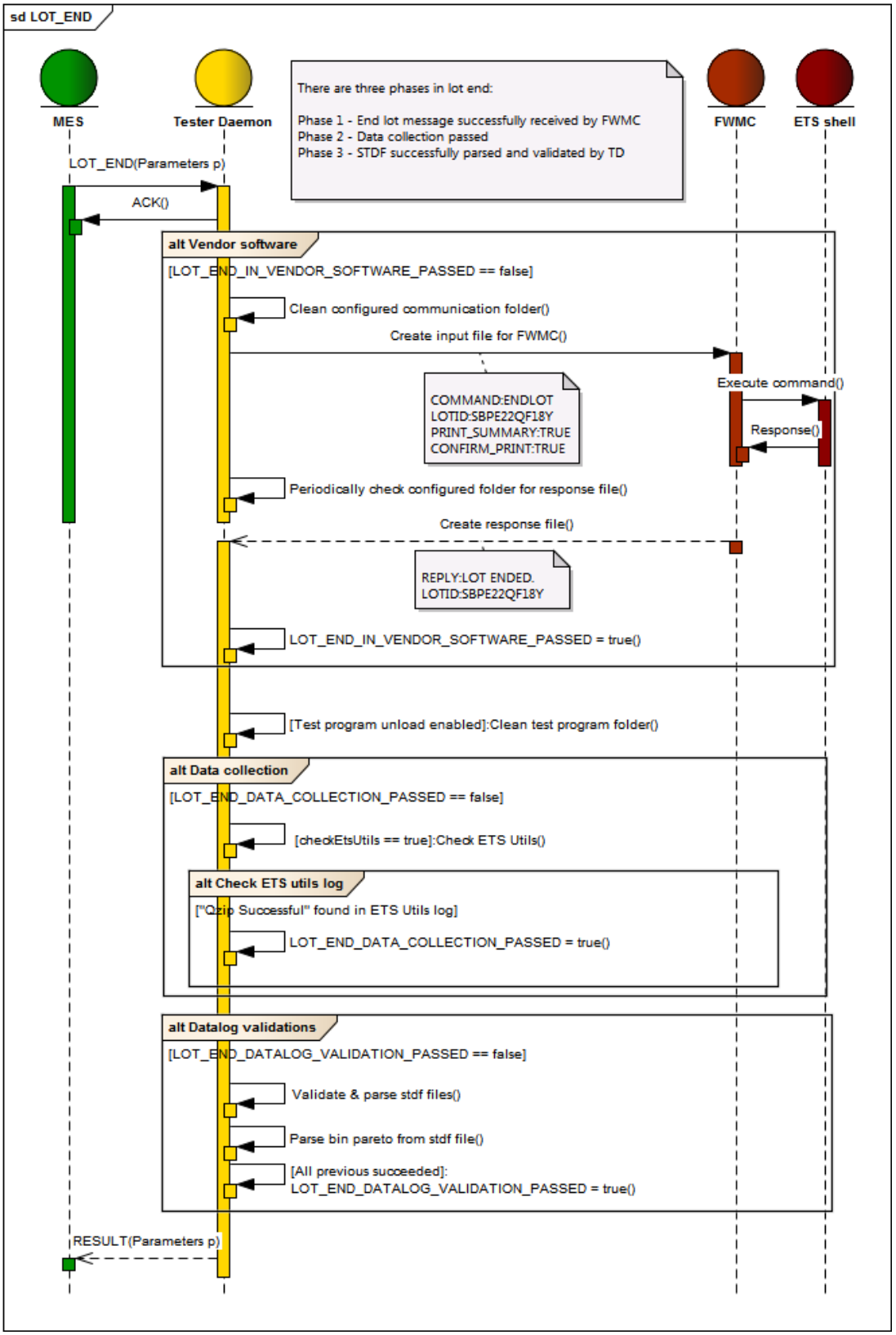
Operace start testování lotu u ovladače pro ETS 364 nejprve ověří, zda program FWMC je spuštěn. V případě, že není spuštěn aplikace jej spustí. Následuje vymazání všech souborů, které obsahuje nakonfigurovaný adresář pro komunikaci s programem FWMC a konstrukce komunikačního souboru dle protokolu pro tento testovací stroj. Následně je v testovacím stroji provedena automatizace zadání identifikátorů lotu, sublotu, typu testovaného objektu, operátoru a automatizace nahrání test programu. V některých případech je nutné provést kalibraci testovacího stroje po nahrání test programu a z tohoto důvodu se může stát, že program FWMC odešle soubor se zprávou o prodloužení časového intervalu pro zpracování žádosti. V tom případě se prodlouží čas na čekání odpovědi před tím, než se zašle chybové hlášení klientovi. Jakmile dorazí soubor s odpovědí o úspěšném vykonání operace, je zaslána odpověď klientovi.



Obrázek 16: Sekvenční diagram operace start testování lotu na ETS 364

Operace konec testování lotu je u ovladače pro ETS 364 prováděna ve třech fázích. První fáze je ukončení testování lotu přes program FWMC. Druhá fáze je validace přesunu datalogu do systému pro sběr dat. Třetí fáze je vnitřní validace datalogu v systému tak, jak bylo již dříve definováno. Rozdělení do tří fází bylo implementováno z důvodu možného opakovaného pokusu o ukončení testování lotu. Pokud již byla provedena některá z uvedených třech fází, tak nebude znovu opakována. Tento problém nemusel být řešen u ovladače pro Powertech QT 6100, kdy znovu ukončení testování lotu nezpůsobuje žádný problém. Scénář o znovu ukončení testování lotu je validní například, když se nepodaří zvalidovat datalogy z měření a klientovi je oznámena chyba. Klient může iniciovat opravu datových souborů a pokusit se znovu o ukončení testování lotu.

První fáze tedy spočívá ve vytvoření komunikačního souboru pro FWMC s žádostí o ukončení testování lotu. V závislosti na požadavku od klienta je pak smazán adresář s test programem. Následuje druhá fáze - ověření, zda se datalog úspěšně dostal do vstupu systému pro sběr dat. Tato akce je prováděna přímo měřícím software „ETS Shell". Jestliže je tento krok proveden úspěšně, je o tomto zapsána zpráva do logu „ETS Shell". Zpráva obsahuje přesný čas této akce a identifikátor měřeného lotu. Pokud je v konfiguračním souboru zapnuta konfigurace „checkEtsUtils", ovladač prohledává log z „ETS Shell" v časovém okně zprávu o úspěšném provedení data kolekce. Třetí, taktéž poslední, fáze je dekódování údajů z měření, vytvoření jednotné formy výstupu a validace údajů v datalogu.



Obrázek 17: Sekvenční diagram operace konec testování lotu na ETS 364

4.6 Testování

V počátcích vývoje ovladače pro Powertechu bylo využito speciálního módu měřícího software pro simulaci testování. Tento způsob módu dovoloval využití externího ovládání testovacího stroje a bylo možné simulovat vstupy/výstupy jednotlivých příkazů. V průběhu testování bylo nutné se několikrát spojit s dodavatelem a zjistit „chování" tohoto módu. Narazil jsem na problém, kdy měřící software nevyprodukoval žádná výstupní data ze simulujícího měření. Bylo zjištěno, že při testování je nutné zaslat do měřícího software speciální příkaz pro vyprodukování simulovaných výstupních dat. Mód pro simulaci měření u testovacího stroje Eagle nebyl příliš spolehlivý a vykazoval značné odchylky od produkčního prostředí.

Pobočka v Malajsii byla vybrána jako první pro pilot nového MES systému, a proto bylo v pozdějších fázích vývoje testováno na reálných testovacích strojích v této lokalitě. Testování probíhalo vzdáleně - byl domluven vzdálený přístup k PC, do kterého byl zapojen a nakonfigurován testovací stroj. Při testování zde probíhala komunikace s technikem, který dle požadavků nastavoval testovací stroj a monitoroval jeho chování. Z počátku se vyskytovaly značné problémy s alokací testovacích strojů. Několikrát se stalo, že testovací stroj v alokovaný čas, určený pro vývoj, nebyl k dispozici a probíhalo zde produkční testování. Situaci se podařilo vyřešit až zásahem vedení z On Semiconductor v Malajsii. Byl zde domluven plán alokující testovací stroje na měsíc dopředu, který již po většinu času dodržen byl.

Ve finální fázi testování byla do Malajsie uskutečněna třítýdenní pracovní cesta, jejíž náplní bylo osobně otestovat ovladače testovacích strojů.

Součástí obecného řešení i ovladačů jsou unit testy. Unit testy jsou implementovány s využitím testovacího frameworku JUnit.

5 Implementační rysy softwarového řešení

Softwarové řešení je implementováno v programovacím jazyce Java ve verzi 1.5 z důvodu zachování kompatibility s operačním systémem Windows NT. Výjimečně je možné se ještě setkat s testovacími stroji (netýká se platforem implementovaných v rámci diplomové práce), které vyžadují Windows NT. Java ve verzi 1.5 je poslední verze, která byla vydána i pro tento systém.

V rámci řešení je využito nástroje Maven pro správu a automatizaci jednotlivých vyhotovení aplikace a aplikačního frameworku Spring. V následujících částech je základní popis těchto nástrojů včetně způsobu využití v implementovaném systému. Informace byly čerpány z oficiálních zdrojů zmíněných nástrojů [4] a [5].

5.1 Apache Maven

Maven je nástroj pro správu a automatizaci buildů. Tento nástroj je využíván napříč aplikacemi vyvíjených ve skupině On Semiconductor. Základem je popis projektu pomocí „Project Object Model", který softwarový projekt popisuje pomocí XML. Je zde popsána závislost na externích knihovnách, včetně umístění těchto knihoven. Nástroj je schopen např. vygenerovat graf závislostí a upozornit na duplicitní závislosti v projektu, to je využíváno zejména u větších projektů. Model popisuje také proces buildování aplikace včetně spuštění testů, přípravu konečného softwarového balíku pro zákazníka nebo přímo nasazení aplikace do produkčního prostředí. Následuje ukázka zkrácené verze modelu pro softwarové řešení diplomové práce:

```
<project>
    <modelVersion>4.0.0</modelVersion>

    <groupId>com.onsemi.cim.apps.testerdaemon</groupId>
    <artifactId>tester-daemon</artifactId>
    <version>2.0.8</version>
    <packaging>jar</packaging>
    <name>tester-daemon</name>

    <url>maven.apache.org</url>

    <parent>
        <artifactId>on-global-parent</artifactId>
        <groupId>com.onsemi.global</groupId>
        <version>1.0.1</version>
    </parent>
```

```
<repositories>
    <repository>
        <id>onsemi-global</id>
        <url>maven.onsemi.com/maven2</url>
    </repository>
</repositories>

<build>
    <plugins>
        <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-compiler-plugin</artifactId>
            <version>2.3.2</version>
            <configuration>
                <source>1.5</source>
                <target>1.5</target>
            </configuration>
        </plugin>
     .
     .
     .

    </plugins>

    <!-- exclude logging configuration & spring context from jar -->
    <resources>
        <resource>
            <directory>src/main/resources</directory>
            <excludes>
                <exclude>**/logback.xml</exclude>
        <exclude>**/AppContext.xml</exclude>
            </excludes>
        </resource>
    </resources>
</build>
<properties>


<dependencies>
```

```
        <dependency>
            <groupId>junit</groupId>
            <artifactId>junit</artifactId>
            <version>4.11</version>
            <scope>test</scope>
        </dependency>
        <dependency>
            <groupId>org.springframework</groupId>
            <artifactId>spring-context</artifactId>
            <version>3.2.8.RELEASE</version>
            <type>jar</type>
        </dependency>
    .
    .
    .
    .

    </dependencies>
</project>
```

Výpis 2: Zkrácená verze pom.xml

Výchozí fáze budování projektu v Mavenu je možno provádět uživatelsky definované akce. Fáze jsou následující: (Názvy stavu byly ponechány v anglickém jazyce, tak aby odpovídaly názvům v modelu)

- **validate**
- **compile**
- **test**
- **package**
- **verify**
- **install**
- **deploy**

V softwarovém řešení je využito tohoto nástroje pro automatické vytvoření jednotného softwarového balení pro zákazníka. Balení vždy obsahuje kořenový adresář s názvem: „*<ovladač testovacího stroje typu>-<verze ovladače>-distribution*", v tomto adresáři je soubor „*run.bat*", kterým je možné aplikaci spustit. Tento adresář obsahuje podadresář „*lib*", který obsahuje všechny

potřebné závislosti pro spuštění a podadresář „*resources*" obsahující konfigurační soubory. Následující definicí v projektovém modelu Mavenu v obecném řešení systému je zajištěno jednotné vytvoření spustitelného softwarového balení i pro budoucí implementované ovladače. Následující definice upravuje proces buildování ve fázi „package", která nastává až po úspěšné kompilaci projektu a testech.

```
<plugins>
        <plugin>
            <artifactId>maven-assembly-plugin</artifactId>
            <version>2.4.1</version>
            <configuration>
                <descriptors>
                    <descriptor>src/assembly/conf.xml</descriptor>
                </descriptors>
            </configuration>
            <executions>
                <execution>
                    <id>make-assembly</id>
                    <phase>package</phase>
                    <goals>
                        <goal>single</goal>
                    </goals>
                </execution>
            </executions>
        </plugin>
         <plugin>
            <groupId>org.apache.maven.plugins</groupId>
            <artifactId>maven-jar-plugin</artifactId>
            <version>2.4</version>
            <configuration>
                <outputDirectory>${project.build.directory}/dist/lib/</
                    outputDirectory>
            </configuration>
         </plugin>
    </plugins>
```

Výpis 3: Využití komponenty „maven-assembly"

5.2 Spring

Spring je aplikační framework pro usnadnění vývoje Java EE aplikací (aplikace určené pro podnikové využití). Celkově se jedná o přibližně 20 komponent, které se kategorizují do těchto modulů:

- **Základní modul Springu - jádro Springu (podpora návrhového vzoru inverze kontroly)**
- **Modul k integraci dat/přístupu k datům**
- **Modul pro vývoj webu**
- **AOP modul pro podporu aspektově orientovaného programování**
- **Modul pro transakční zpracování**
- **Modul pro podporu testování**

V rámci diplomové práce je využit základní modul Springu. Tento modul obsahuje implementaci návrhového vzoru inverze kontroly („Inversion of Control"). Tento návrhový vzor funguje na principu přenesení zodpovědnosti za vytvoření instancí a provázání objektů z aplikace na aplikační framework.

Vkládání závislostí („dependency injection") je speciální případ inverze kontroly a řeší způsob vložení závislostí mezi jednotlivými komponentami aplikace tak, aby jedna komponenta mohla používat druhou aniž by na ni měla v době kompilace referenci. Jednotlivé závislosti tedy nejsou známy v době kompilace aplikace, ale až v době jejího spuštění. Objekty jsou aplikačním frameworkem vytvořeny na základě konfiguračního souboru ve formátu XML. Následuje ukázka konfiguračního souboru XML:

```
.
.
    <bean id="TesterDaemon" class="com.onsemi.cim.apps.testerdaemon.
        TesterDaemon" >
        <property name="commandTimeout" value="300" />
    </bean>

    <bean id="Connector" class="com.onsemi.cim.apps.testerdaemon.connector.
        TcpIpConnector" >
        <property name="port" value="5000" />
    </bean>
```

```
    <bean id="TesterDriver" class="com.onsemi.cim.apps.testerdaemon.
        testerdriver.PowertechDriver">
        <property name="powertechDriverPort" value="6000" />

    </bean>
.
.
```

Výpis 4: Část souboru AppContext.xml - vkládání závislostí pomocí Springu

Tímto způsobem jsou řešeny všechny konfigurační položky ovladačů testovacích strojů i samotné nastavení konkrétní implementace ovladače. V budoucnu bude stačit pouze implementovat dané rozhraní „TesterDriver" a pomocí vložení závislostí je komponenta ovladače vložena do obecného systému. Podobným způsobem je také řešena konkrétní implementace transportní vrstvy mezi systémem a klientem. Ve výše zmíněné ukázce je dále zobrazená konfigurace jednotlivých TCP komunikačních portů a hodnota čekání v sekundách na provedení příkazu před nahlášením chyby.

6 Závěr

V rámci diplomové práce vzniklo modulární softwarové řešení, složené z obecné části a ovladače, jehož jedna instance řeší automatizaci a integraci jednoho typu testovacího stroje. Výstupem práce je úspěšná implementace ovladačů pro dva typy testovacích strojů a implementace obecného řešení, jenž bude základem pro automatizaci dalších typů testovacích strojů.

V Malajsii proběhl úspěšný pilot nového MES systému, při němž byl využit ovladač pro testovací stroj ETS 364. Projekt nasazení nového MES systému stále ještě probíhá a ladí se detaily ve výrobních procesech před „final testem". Z tohoto důvodu softwarové řešení vyvinuté v rámci diplomové práce ještě není nasazeno v produkci.

S implementací ovladače pro testovací stroj ETS 364 se počítá do projektu „Mapper for automotive". Projekt se zabývá úpravou softwarového řešení „Mapper", které je použito v oblasti testování desek tak, aby jej bylo možné certifikovat a použít na testování produktů pro automobilový průmysl. Přínosem ovladače pro projekt bude automatizace nahrávání test programu, validace kontrolního součtu test programu a validace měřených dat.

Očekává se, že stávající obecné řešení, včetně již implementovaných ovladačů, bude dále vyvíjeno dle požadavků interních zákazníků. Stávající řešení pokrývá automatizaci dvou platforem testovacích strojů. Plán do budoucna je využít výstup této diplomové práce a pokračovat na vývoji dalších ovladačů tak, aby byly pokryty všechny možné (automatizovatelné) platformy testovacích strojů ve společnosti On Semiconductor.

Literatura

[1] PÁNEK, Petr a kolektiv. *Základy technologie výroby polovodičů.* Výdáno 2016. ISBN 978-80-7204-939-4.

[2] *Hi-Speed Discrete Test System QT-6000.* [online] [cit. 2017-5-2].
URL: <http://www.powertechsemi.com/Test_System/QT_6000_Series/21.html>

[3] *ETS-364 / ETS-600 Test Systems.* [online] [cit. 2017-10-2].
URL: <http://www.teradyne.com/products/semiconductor-test/
ets-364-ets-600-test-systems>

[4] *Maven - Introduction to the Build Lifecycle.* [online] [cit. 2017-15-3].
URL: <http://maven.apache.org/guides/introduction/introduction-to-the-lifecycle.html>

[5] *Spring - Documentation.* [online] [cit. 2017-17-3].
URL: <https://spring.io/docs/reference>

A Příloha na CD

Příloha na CD obsahuje softwarové řešení vyvinuté v rámci diplomové práce. Přesný popis struktury přílohy na CD je možné nalézt v souboru „*readme.txt*".